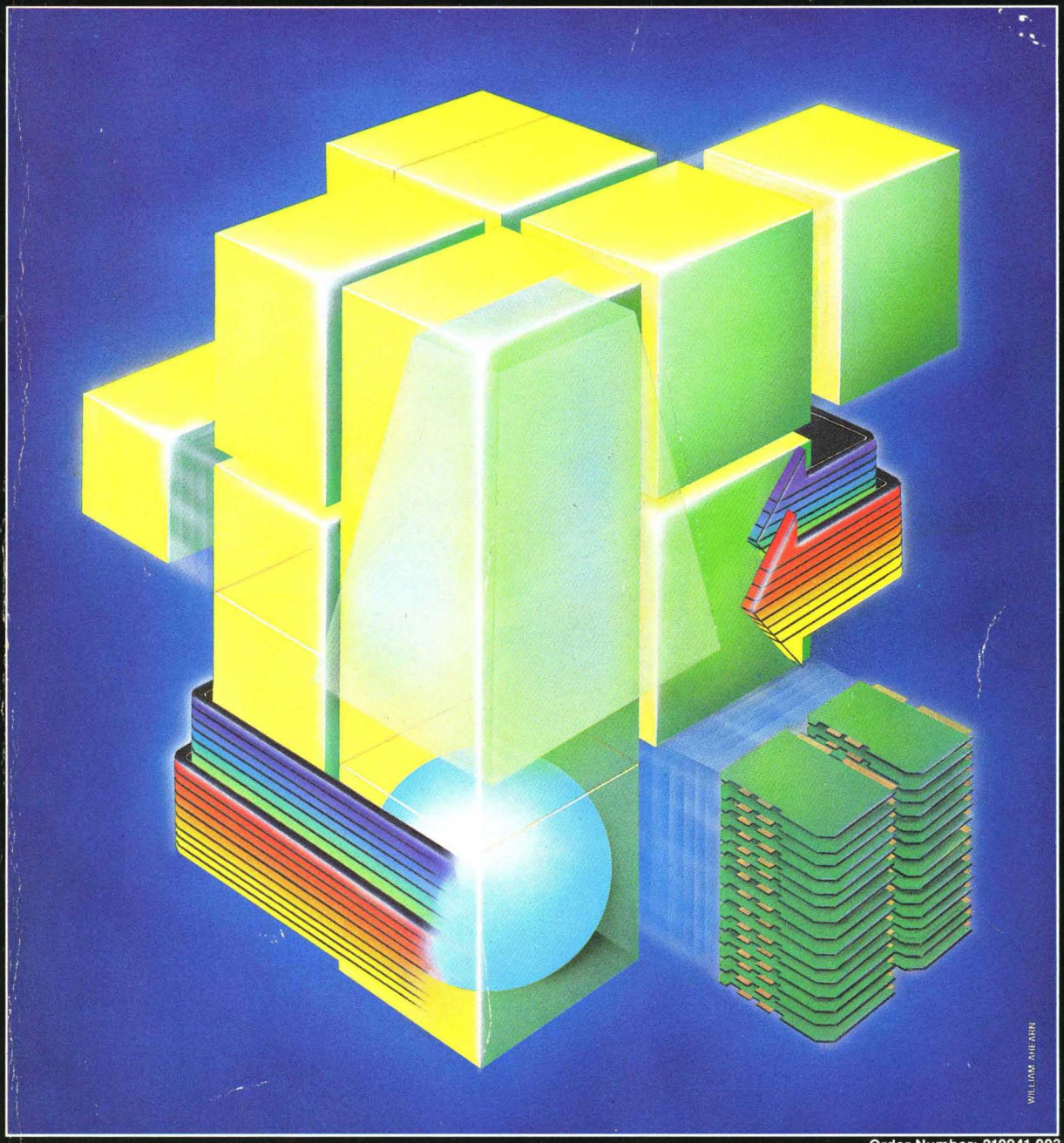
intel

OEM Systems Handbook



Order Number: 210941-003

LITERATURE

In addition to the product line handbooks listed below, the INTEL PRODUCT GUIDE (no charge, Order No. 210846-003) provides an overview of Intel's complete product lines and customer services.

Consult the INTEL LITERATURE GUIDE (Order No. 210620) for a listing of Intel literature. TO ORDER literature in the U.S., write or call the INTEL LITERATURE DEPARTMENT, 3065 Bowers Avenue, Santa Clara, CA 95051, (800) 538-1876, or (800) 672-1833 (California only). TO ORDER literature from international locations, contact the nearest Intel sales office or distributor (see listings in the back of most any Intel literature).

Use the order blank on the facing page or call our TOLL FREE number listed above to order literature. Remember to add your local sales tax.

1985 HANDBOOKS

Product line handbooks contain data sheets, application notes, article reprints and other design information.

	*U.S. PRICE
QUALITY/RELIABILITY HANDBOOK (Order No. 210997-001) Contains technical details of both quality and reliability programs and principles.	**$15.00**
CHMOS HANDBOOK (Order No. 290005-001) Contains data sheets only on all microprocessor, peripheral, microcontroller and memory CHMOS components.	**$12.00**
MEMORY COMPONENTS HANDBOOK (Order No. 210830-004)	**$18.00**
TELECOMMUNICATION PRODUCTS HANDBOOK (Order No. 230730-003)	**$12.00**
MICROCONTROLLER HANDBOOK (Order No. 210918-003)	**$18.00**
MICROSYSTEM COMPONENTS HANDBOOK (Order No. 230843-002) Microprocessors and peripherals—2 Volume Set	**$25.00**
DEVELOPMENT SYSTEMS HANDBOOK (Order No. 210940-003)	**$15.00**
OEM SYSTEMS HANDBOOK (Order No. 210941-003)	**$18.00**
SOFTWARE HANDBOOK (Order No. 230786-002)	**$12.00**
MILITARY HANDBOOK (Order No. 210461-003) Not available until June.	**$15.00**
COMPLETE SET OF HANDBOOKS (Order No. 231003-002) Get a 25% discount off the retail price of $160.	**$120.00**

***U.S. Price Only**

intel®

U.S. LITERATURE ORDER FORM

NAME: _____ TITLE: _____

COMPANY: _____

ADDRESS: _____

CITY: _____ STATE: _____ ZIP: _____

COUNTRY: _____

PHONE NO.: (____) _____

ORDER NO.	TITLE	QTY.	PRICE	TOTAL
☐☐☐☐☐☐-☐☐☐		____	× ____	= ____
☐☐☐☐☐☐-☐☐☐		____	× ____	= ____
☐☐☐☐☐☐-☐☐☐		____	× ____	= ____
☐☐☐☐☐☐-☐☐☐		____	× ____	= ____
☐☐☐☐☐☐-☐☐☐		____	× ____	= ____
☐☐☐☐☐☐-☐☐☐		____	× ____	= ____

POSTAGE AND HANDLING:
Add appropriate postage and handling to subtotal
10% U.S.
20% Canada

Subtotal _____

Your Local Sales Tax _____

_____

Allow 4-6 weeks for delivery

Total _____

Pay by Visa, MasterCard, Check or Money Order, payable to Intel Literature. Purchase Orders have a $50.00 minimum.

☐ Visa ☐ MasterCard Account No. _____ Expiration Date _____

Signature: _____

Mail To: Intel Literature Distribution
Mail Stop SC6-714
3065 Bowers Avenue
Santa Clara, CA 95051.

Customers outside the U.S. and Canada should contact the local Intel Sales Office or Distributor listed in the back of this book.

For information on quantity discounts, call the 800 number below:
TOLL-FREE NUMBER: (800) 548-4725
Prices good until 12/31/85.
Source HB

Cut Along Dotted Line

For a free subscription to the European edition of *SOLUTIONS* please mail this card to one of the regional offices listed below:

Intel Corporation (UK) Ltd
Piper's Way
Swindon, SN3 1RJ
Wiltshire, England

Intel Semiconductor GmbH
Seidlstrasse 27
D-8000 Munchen 2
West Germany

Intel Corporation S.A.R.L.
5 Place de la Balance
Silic 223
94528 Rungis Cedex
France

Intel Sweden AB
Box 20092
Archimedesvagen 5
S-16120 Bromma
Sweden

Intel Corporation Italia Spa
Milanofiori, Palazzo E
20094 Assago (Milano)
Italy

intel

OEM SYSTEMS HANDBOOK

1985

About The Cover:
The cover design is an abstract portrayal of Intel's computer systems products as the building blocks of systems development. The building blocks meld together, encompassing the heart of an open system, which is symbolized by the pyramid at the heart of the architecture. This melding or flexibility is the key Intel provides to the design engineer for development of MULTIBUS® systems architecture.

Intel Corporation makes no warranty for the use of its products and assumes no responsibility for any errors which may appear in this document nor does it make a commitment to update the information contained herein.

Intel retains the right to make changes to these specifications at any time, without notice.

Contact your local sales office to obtain the latest specifications before placing your order.

The following are trademarks of Intel Corporation and may only be used to identify Intel Products:

BITBUS, COMMputer, CREDIT, Data Pipeline, GENIUS, i, $\hat{\text{i}}$, ICE, iCS, iDBP, iDIS, I^2ICE, iLBX, i_m, iMDDX, iMMX, Insite, Intel, int_el, int_elBOS, Intelevision, int_eligent Identifier, int_eligent Programming, Intellec, Intellink, iOSP, iPDS, iRMX, iSBC, iSBX, iSDM, iSXM, KEPROM, Library Manager, MCS, Megachassis, MICROMAINFRAME, MULTIBUS, MULTICHANNEL, MULTIMODULE, OpeNET, Plug-A-Bubble, PROMPT, Promware, QUEST, QueX, Ripplemode, RMX/80, RUPI, Seamless, SLD, SYSTEM 2000, and UPI, and the combination of ICE, iCS, iRMX, iSBC, iSBX, MCS, or UPI and a numerical suffix.

MDS is an ordering code only and is not used as a product name or trademark. MDS® is a registered trademark of Mohawk Data Sciences Corporation.

* MULTIBUS is a patented Intel bus.

Additional copies of this manual or other Intel literature may be obtained from:

Intel Corporation
Literature Department
3065 Bowers Avenue
Santa Clara, CA 95051

Table of Contents

*XENIX is a trademark of Microsoft Corporation

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Alphanumeric Index

Alphanumeric Index

Alphanumeric Index

Integrated Microcomputer Systems 1

PRELIMINARY

iSXM™ SERIES
SYSTEM EXTENSION MODULES

- **iSXM™ 101 Memory Management and Protection Extension Module for XENIX†-86 multi-user microcomputer systems**
- **iSXM™ 534 Four-channel Serial Communication Extension Module for multi-user systems**
- **iSMX™ 544 and iSXM™ 544A Intelligent Four-channel Serial Communication Extension Modules for high-performance multi-user systems**
- **iSXM™ 951 RS232 Serial I/O Cable Extension Module for use with iSXM 534 or 544 in Intel SYSTEM 310 Family microcomputer systems**
- **iSXM™ 953 RS232 Serial I/O Cable Extension Module for use with iSXM™ 534 or 544 in Intel SYSTEM 380 Family microcomputer systems, or as a second cable set for the SYSTEM 86/330A**
- **iSXM™ 955 Communication Cabling for use with the iSBC® 188/48 Eight Channel Communication board in the SYSTEM 310 Family**
- **iSXM™ 956 Communication Cabling for use with the iSBC® 188/48 Eight Channel communication board in the SYSTEM 380 Family**

The iSXM™ family of System Extension Modules is designed to extend the hardware capability of the SYSTEM 86/300 and SYSTEM 286/300 Series microcomputers. All hardware is fully configured and can easily be installed in the system. An easy-to-follow installation manual as well as all hardware documentation is included in each package.

† XENIX is a trademark of Microsoft Corporation.

OCTOBER 1984
ORDER NUMBER: 230877-002

iSXM™ 101 XENIX† Memory Management and Protection Module

This module contains the iSBC® 309 Memory Management and Protection MULTIMODULE™ board, the iAPX 8087 Numeric Data Processor Device, all associated mounting hardware and an installation guide for use in the SYSTEM 86/300 Series microcomputers.

XENIX†-86 takes advantage of these hardware features through the use of a dynamic scatter loading technique. This memory management method allows programs to be loaded in non-contiguous blocks of RAM for more efficient memory utilization.

XENIX-86 Release 2.0 supports the 8087, giving the host processor greatly enhanced math execution capability.

The two-level diagnostic program featured in Intel SYSTEM 86/300 Series microcomputers supports the iSXM™ 101.

For more details, please refer to the iSBC 309 and iAPX 8087 data sheets.

iSXM™ 534 Four-Channel I/O Extension Module, iSXM™ 544 and iSXM™ 544A Intelligent Four-Channel I/O Extension Module

The iSXM 534 contains the iSBC 534 Four-channel Serial Communications Board, fully configured for use in Intel SYSTEM 300 Series microcomputer systems, installation instructions and a Hardware Reference Manual.

The iSXM 544 Intelligent Four-Channel Serial I/O Module contains the iSBC 544 board, fully configured for use in Intel SYSTEM 300 Series microcomputer systems, two 2732A EPROMS containing firmware to control the iSBC 544, installation instructions and Hardware Reference Manual.

Installation of these modules adds four serial I/O channels to the one already resident on the 86/30 processor board, or to the two resident on the 286/10 processor board. They interface directly to the system through the MULTIBUS® system bus. The four serial ports fully support RS232C (configured) asynchronous communications.

The iSXM 544 provides much higher I/O performance, making it particularly suited for multiuser requirements.

The iSXM 554A is functionally equivalent to the iSXM 544 and offers 16 MB addressability.

These System Extension Modules are fully supported by iRMX™ 86 Release 5, XENIX 86 Release 2 and XENIX 286.

Cables and mounting hardware are required for use in Intel microcomputer systems. Order the iSXM 951, 952, or 953 depending on your system type.

For a full explanation of the iSBC 534 or iSBC 544, please refer to the respective data sheets.

iSXM™ 951 RS232 Serial I/O Cables

This module contains four 1-foot cables for use from the edge connectors of the iSXM 534 or 544 to the back of the SYSTEM 86/310 Series or SYSTEM 286/310 Series chassis. An installation guide is also included.

iSXM™ 953 RS232 Serial I/O Cables

This module contains four 2-foot cables for use from the edge connectors of the iSXM 534 or 544 to the back of the SYSTEM 86/380 Series or SYSTEM 286/380 Series chassis, and an installation guide.

Reference Manuals

173077-001	iSXM 101 Installation Guide
144686	iSBC 309 Hardware Reference Manual
173177-001	System Terminal Communication Installation Guide
980450	iSBC 534 Hardware Reference Manual
980616	iSBC 544 Hardware Reference Manual
173074-001	iSXM 951 Installation Guide
173076-001	iSXM 953 Installation Guide

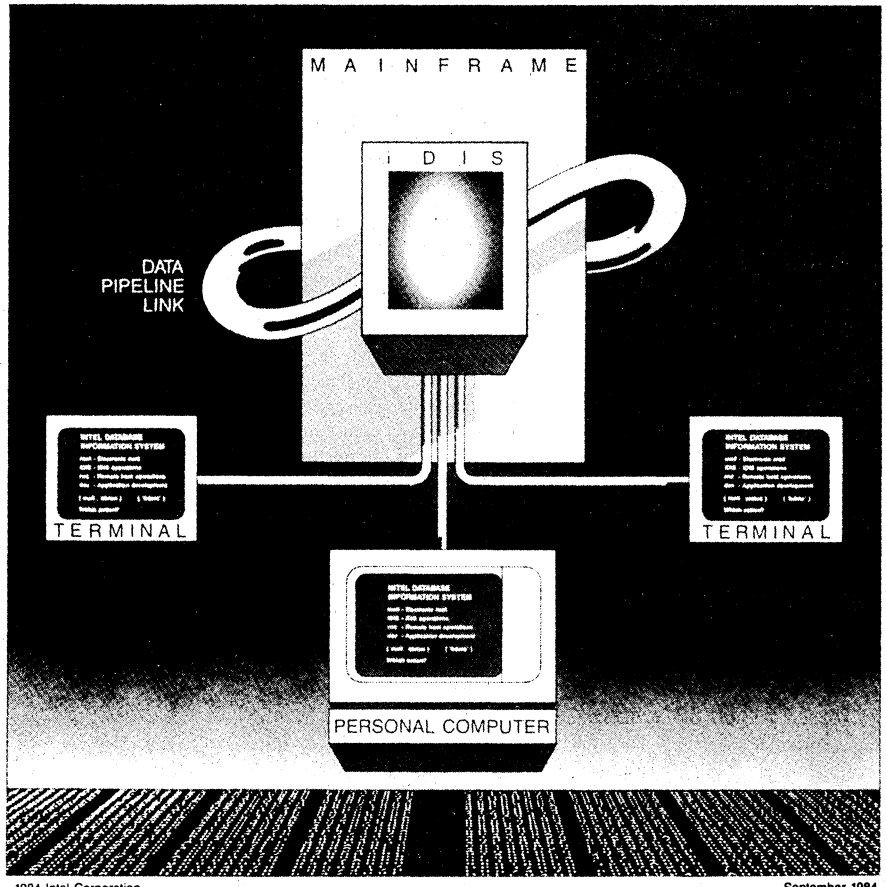
Ordering Information

iSXM 101 or PSXM 101	Memory Management Module
iSXM 534 or PSXM 534	4-channel I/O Extension Module
iSXM 544 or PSXM 544	Intelligent 4-channel I/O Extension Module
iSXM 544A or PSXM 544A	16 MB Intelligent 4 Channel I/O Extension Module
iSXM 951 or PSXM 951	Cables for SYSTEM 86/310, SYSTEM 286/310
iSXM 953 or PSXM 953	Cables for SYSTEM 86/380, SYSTEM 286/380
iSXM 955 or PSXM 955	Cables for the iSBC 188/48 in any SYSTEM 310
iSXM 956 or PSXM 956	Cables for the iSBC 188/48 in any SYSTEM 380

Database Information System iDIS™ 715

- Building block for departmental-level applications
- Data Pipeline™ system to distribute databases
- Direct mainframe database extract and file transfer facilities
- Gateway for personal computer and terminal access
- Multiuser XENIX* 3.0 operating system
- Local relational database management and report writer
- Integrated software with on-line help facility
- Word processing, spreadsheet, graphics, menu development, and communication options
- C programming language
- Desk-top integrated microsystem
- Worldwide vendor service and support

*XENIX is a registered trademark of Microsoft Corporation.



September 1984
ORDER NUMBER 210712-005

Building Vertical Applications with the iDIS™ System

The Intel Database Information System (iDIS™) is a fully-integrated multiuser hardware/software microcomputer system. It serves as a building block for end-user applications and a powerful access tool in the Data Pipeline connection between a mainframe and the end-user. Data can be maintained by central data processing departments and distributed to departmental users through a network of terminals and PCs. The system can be configured as a gateway in the micro-to-mainframe flow of data or as a stand-alone processor with shared local database capabilities. The iDIS system includes an SQL-compatible, multiuser relational DBMS for shared access to disk storage and features a full range of information processing functions for multiple concurrent users at all levels of technical skill.

The iXTRACT remote database extract facilities

The iDIS system offers two interactive, menu-driven modes of database extract. With the Remote File Transfer (RFT) iXTRACT facility, a "flat file" (sequential) data structure can be downloaded from the mainframe and converted into a local relational database. Using host computer utilities to generate the flat file, the RFT facility can download data from virtually any DBMS or file management system. The facility is bidirectional, such that flat files can be transmitted between a mainframe host and an iDIS system with its network of terminals and personal computers.

A second facility, the Direct iXTRACT facility, is a menu-driven data extract facility which directly downloads Intel SYSTEM 2000® databases (from IBM, CDC, and Sperry environments) into an iDIS database. Both RFT and direct modes allow non-technical users to access remote corporate databases and extract information while central data processing controls data security at every terminal.

*UNIX is a trademark of AT&T Bell Laboratories. Multiplan is a registered trademark of Microsoft Corporation.

Microsoft XENIX

The iDIS operating system is provided by XENIX 3.0, an enhanced industry-standard version of UNIX* XENIX is a general-purpose, multiuser, interactive operating system designed to make the computing environment simple, efficient and productive for a wide range of users. While the system developer has access to all XENIX functionality, the operating system appears to be transparent to the user who interacts with the iDIS software through its menu system.

The XENIX system supplies:

- A flexible and logical hierarchical file system, with cross-directory file linking and multiple protection and security modes
- The XENIX shell command language, with conditional, recursive, and iterative constructs (for development of user/application procedures)
- Sequential, asynchronous, and background process execution
- Sophisticated editing and text-processing facilities supporting printers and typesetters
- Device-independent input and output.

The iDB-local relational database management (DBMS)

The iDIS system offers the iDB DBMS, a full-function relational DBMS that supports an interactive query/update language similar to that of IBM's SQL. Included with iDB is a Report Writer package. This allows users to prepare custom reports quickly from information in iDB without programming knowledge. The iDB DBMS offers all the power of a mainframe DBMS at the microsystem level. Multiple iDB users can concurrently access common local databases with confidence in system integrity.

Other features include:

- A user-prompting data entry and update subsystem
- A bulk loading and unloading utility for rapid transfer of data among files and databases
- Extensive on-line help facilities
- Descriptive error and diagnostic messages
- Programmatic interface to the C language and XENIX shell.

Seamless™ software interface

The iDIS software family is integrated into a Seamless set of productivity tools. Data can be easily transferred among the various iDIS application packages, such that the iWORD processor, iPLAN (Multiplan*) spreadsheet, and iDB DBMS can interchange data and reports. All iDIS decision-support tools can be easily brought to bear on a particular data-analysis problem.

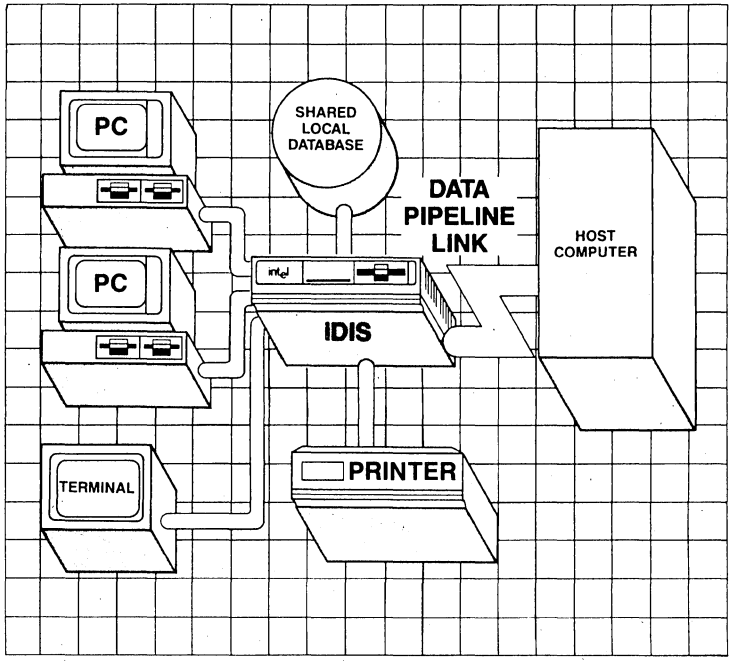
Individually, each package is accessible through a common user interface – a hierarchical menu system serving as a superstructure for the complete iDIS system. A common help facility binds all iDIS software.

The iWORD processor

The Intel iWORD facility is a sophisticated word processing tool that supports a complete office-wide range of document preparation functions. The iWORD user can develop, edit, store, format and print a variety of presentation-quality business documents, including reports, memoranda, technical documents, specifications and manuals. All iWORD commands are in plain English and many can be executed by a single keystroke. The iWORD processor is also sufficiently powerful for the experienced user, offering access to XENIX text processing capabilities including the printer- and typesetter-drivers nroff and troff. An on-line help facility is continuously available.

Major iWORD editing and formatting features include:

- Full-screen editor with on-line display of formatted text
- Embedded commands for global formatting
- Spelling/dictionary module and mail/merge facility
- Right justification, underlining, indentation, centering, footnotes, superscripts and subscripts.



The iPLAN (Multiplan) spreadsheet

The iDIS system supports 'what if' decision-modeling with iPLAN (Multiplan) Spreadsheet, a multi-purpose tool capable of a wide variety of business and scientific tabulations. The iPLAN user can custom-tailor a versatile two-dimensional matrix for specific analyses, including financial modeling, planning and forecasting. Like the other functions in the iDIS software environment, the iPLAN spreadsheet accepts data four ways: from the keyboard, from the iDB DBMS, from mainframe databases (via the data extract facility), and from formatted XENIX files.

Important iPLAN features include:

- Easy-to-use English commands
- Vertical and horizontal scrolling, multi-window and multi-table display
- Presentation of extra large tables
- Linking and updating multiple interrelated spreadsheets
- Automatic updating of calculations
- Alphanumeric sorting capabilities
- Extensive, on-line help facility.

The personal computer (iPC) connection

To complete the Data Pipeline connection, the iDIS system offers a menu-driven file conversion and transfer facility that allows single-user PC files to be accessed in the multiuser XENIX environment. The PC user can use the iDIS system to convert database and spreadsheet files from popular PC file formats (such as dBASE II* Lotus 1-2-3* and Multiplan formats) to iDB file formats. As a result, mainframe files can be downloaded to relational structures within iDB databases and further converted and downloaded to PC-based files for local applications analysis. The PC user can operate in three modes: bidirectional iDIS-to-PC file transfers, iDIS terminal emulation, and local PC-DOS control.

Office automation features

The XENIX operating system provides an electronic mail service in which business messages are shared and relayed with ease. XENIX also includes handy "desk calculator" functions and an electronic calendar that provides an automatic reminder (via electronic mail) of any user appointments.

Application development tools

The iDIS application development subsystem includes such software tools as the iMENU development system, C programming language, the XENIX Shell program, and the full-screen 'vi' editor (visual editor). These tools support efficient development and maintenance of program and text files by technical users.

The iDIS system offers a complete program development and execution environment for C, the versatile general-purpose language in which the operating system and all iDIS application packages are implemented. C maximizes development productivity by its structured programming methodologies and standard flow-control constructions—if, while, for, do, and switch (case). It provides pointers, the ability to perform address arithmetic, and recursive functions. Many existing C-based applications can be efficiently ported to the iDIS system.

The iMENU development facility

The iMENU development facility provides the iDIS system-level user interface, tying together the XENIX operating system, iDIS applications software, and help system. The iMENU facility retains and yet simplifies full XENIX functionality. Programmers and non-programmers alike can use the iMENU facility in creating or modifying menus, forms, and help screens for existing or custom-developed applications.

The on-line help facility

The help facility, a comprehensive on-line documentation feature, is integrated with the menu system so the user need not refer to hard copy reference manuals when using iDIS applications. Experienced users can employ the iMENU facility to extend or modify the help facility to specify help procedures for custom applications.

The iGRAPH facility

The iDIS system offers a presentation graphics package, iGRAPH, that provides high quality output to most

*dBASE II is a trademark of Ashton-Tate.
Lotus and 1-2-3 are trademarks of Lotus Development Corporation.

standard graphics peripherals. Through iDIS integration, data can easily be moved from iPLAN and iDB to iGRAPH, from iGRAPH to iWORD for printed output, and to and from iPC. Graphic peripherals supported are Tektronix 4105, color terminal, Televideo 950 monochrome terminal with Retrographics board, Hewlett-Packard HP7475 plotter, the Epson MX80 printer, and the IBM PC with graphics board. The Intel terminal can be used to generate graphics hard copy.

Communications

The iDIS communications subsystem provides remote job entry (RJE) to mainframe hosts through its emulation of a HASP multileaving workstation or 2780/3780 protocol. TTY passthrough facilities also provide direct access to remote interactive applications, including other iDIS systems and personal computers. Support for 3270 BCS emulation is also available, and SNA support is planned.

BASE SYSTEM HARDWARE

Processor

The iDIS 715 uses the MULTIBUS®-based iSBC® 286/10 board with the 80286 processor. An Intel 80287 coprocessor is standard to provide significant performance boost for numeric operations. Instructions are 8, 16, or 32 bits in length; data are 8 or 16 bits long; numeric processing, with the 80287, is carried out in 80-bit words. Memory management and protection are also included.

One megabyte of high-performance RAM with ECC is standard. To provide faster access, Intel memory boards are connected directly to the 286/10 processor board via iLBX™ (Local Bus Exchange).

Communications support

Asynchronous communication and synchronous mainframe communications support is handled by an iSBC 188/48 Advanced Communications Processor Board. Eight connections can be configured for terminals, PCs and/or mainframe communications.

MASS STORAGE

Winchester disk drive—The iDIS 715 contains a 40 MB 5¼" Winchester technology disk drive for program and data storage. The drive has an average access time about 40 milliseconds and a transfer rate of 6.44 Mbits/sec.

Intelligent controller—The iDIS 715 includes an intelligent, 8089-based iSBC 215 Winchester controller. This high performance interface contains firmware which is executed directly on the iSBC 215 controller to offload a significant portion of disk I/O overhead from the host 80286 processor. In addition, the iSBC 215 board supports an iSBX™ 218 controller to manage the floppy disk.

Floppy disk drive—A 5¼" 320 KB floppy disk drive is included in the base system. This floppy drive has an average access time of 91 milliseconds and a transfer rate of 250 KB/sec.

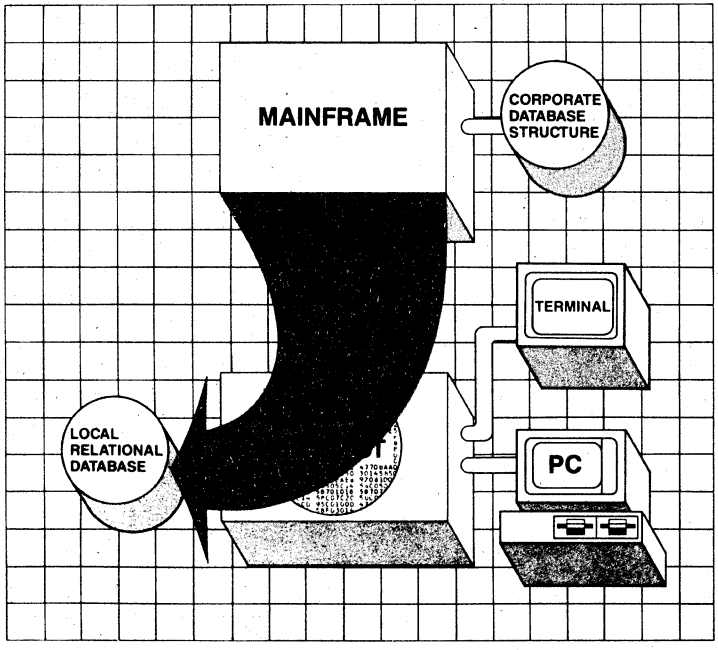
Optional peripherals

Display terminal—The standard iDIS system can connect up to 10 terminals. The terminal connected to the console port can be used for system control and administration, as well as a regular workstation. An iDIS terminal is also available, with 24-80 character lines and a 25th status line. The 96 upper and lower ASCII characters are formed in a 7x9 matrix in an 8x10 cell. The screen is non-reflective and uses green P31 phosphor.

Printer—The iDIS system supports a Centronics-compatible printer. A dot matrix character printer capable of 200 character-per-second bidirectional printing is also available. It includes a Centronics-compatible parallel interface and a 218-character buffer. The printer features a 7×9 dot matrix to form all 96 ASCII characters. Maximum standard print line length is 132 characters or 218 with condensed print. The Intel printer prints 10 characters per inch; spacing of 5, 8.25, and 16.5 characters per inch can also be accommodated.

System support

The iDIS system is fully supported by Intel's worldwide service staff, including a group of information system



professionals with over 15 years experience in commercial database technologies. Support for iDIS applications software is included with the system price for 90 days and is optional thereafter. All iDIS software with a current software maintenance agreement is supported by the Intel Austin Systems Support Hotline.

In addition to the hotline, system support includes software updates, customer problem reporting, and a product newsletter. Intel provides comprehensive training classes on all iDIS applications, the XENIX operating system, programming languages, and hardware operation.

The iDIS hardware includes a warranty for 90 days mechanical, 45 days labor, and 90 days electrical components. After the warranty period, subscription maintenance is available from the Intel field service organization. Hardware service is also available for users on a per-call basis.

Extensive system documentation

The iDIS 715 is shipped with multiple hardware and software manuals that address all aspects of system operations. Software documentation includes manuals on the XENIX Release 3.0 operating system, iDB and Report Writer software, and each optional application package that is ordered. General overviews and detailed tutorials are an integral part of this documentation. A system installation and maintenance manual, a system overview manual, and a site preparation manual are also provided.

SYSTEM CONFIGURATION
Base Hardware System:

- 1 MB of RAM memory
- 320 KB floppy drive
- 40 MB Winchester disk
- Support for up to 10 terminals and/or PCs
- Printer support for Centronics-compatible printer
- Disk controller board
- Communications processor

Optional Hardware:

- Additional communications processor
- Terminals and dot-matrix printers

Base Software System:

- XENIX 3.0 operating system
- C programming language
- 'vi' editor
- XENIX utilities (including nroff and troff text processors)
- Electronic mail and calendar
- iDB and Report Writer
- iMENU (runtime) system
- Help facility
- Complete systems diagnostics

Optional Software:

- iWORD word processor
- iPLAN (Multiplan) spreadsheet
- iMENU menu development system
- Direct iXTRACT facility
- Remote File Transfer facility
- iPC (personal computer link)
- iGRAPH presentation graphics
- RJE communication support (2780/3780 and/or HASP protocol)
- 3270 BSC emulation

SPECIFICATIONS
Instruction cycle time

250 nanoseconds for fastest executable instructions.

Disk

Standard 40 MB Winchester disk; Second 40 MB disk is planned.

External/PC interface

Serial – 8 asynchronous ports, configurable from 110 to 9600 baud. EIA Standard RS232C signal support is provided.

Parallel – one Centronics-compatible parallel I/O port for printer connections.

Regulatory Agency Specifications

Meets UL114-Safety; CSA 22.2-Safety; FCC Docket 20780-RFI/EMI. Designed to meet IEC 435-Safety; VDE 0871-RFI/EMI.

ENVIRONMENTAL OPERATING REQUIREMENTS

Altitude – Sea level to 8000 feet.

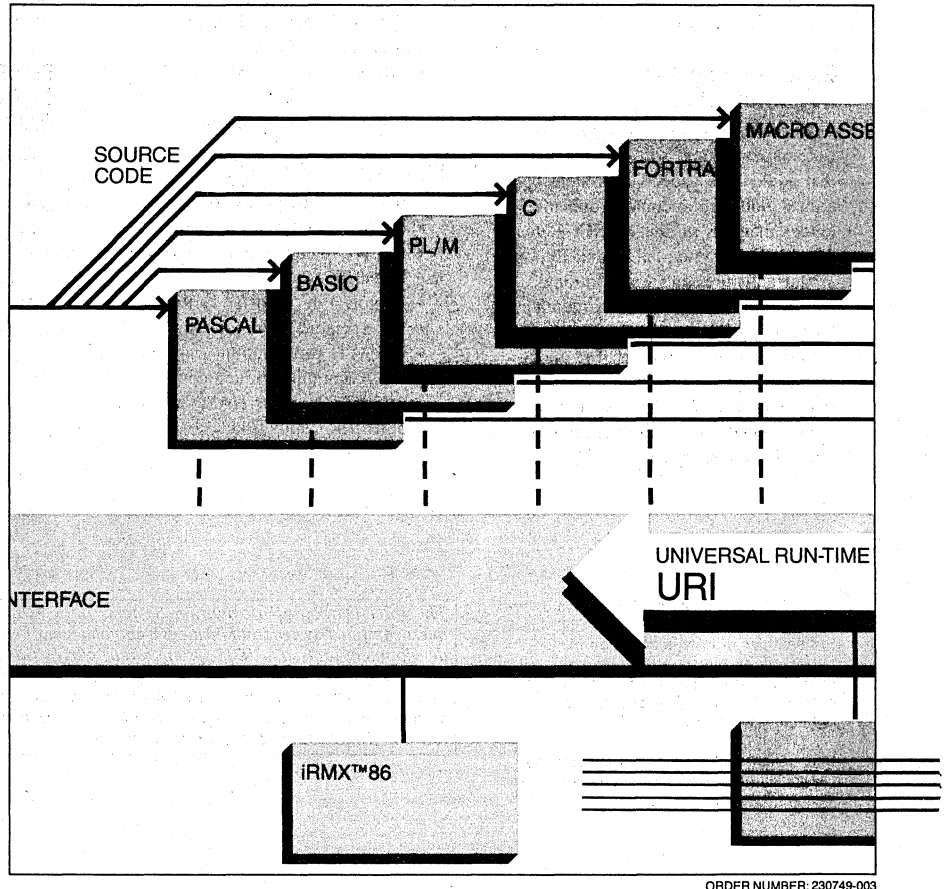
Temperature – 15 degrees C to 35 degrees C.

Relative humidity – 20% to 80% non-condensing over the operating temperature range. The environmental combination of humidity and temperature together cannot exceed 26 degrees C wet bulb.

The Intel Database Information System (iDIS 715) is a multiuser XENIX 286-based microcomputer system that includes a complete set of end-user productivity and application development tools. The iDIS system can be purchased as an integrated hardware/software microcomputer system configured for specific departmental computing applications. In addition, the various iDIS productivity tools, application development tools, relational database management system, and communication software options can be purchased separately to run on XENIX-based Intel microcomputers.

iRMX™ LANGUAGE

- **Industry-standard languages and utilities for developing applications on iRMX-based systems. Includes FORTRAN, Pascal, C, BASIC, PL/M, assembler, text editor**
- **Complete set of utilities to create and manage object modules**
- **Mix languages on single application system with UDI standard**
- **Intel 8087 and 80287 math coprocessor support**
- **8086 and 80286 compatibility**
- **Worldwide post-sales service and support organization**



*XENIX is a trademark of Microsoft Corporation.
†UNIX is a trademark of Bell Laboratories.

ORDER NUMBER: 230749-003

Full Language Support for iRMX™-Based Systems

Intel's iRMX™ 86 based systems are completely supported by a wide variety of popular languages and utilities with which to build fast, real-time, multi-tasking applications. Included are the latest versions of FORTRAN, Pascal, BASIC, C, PL/M and Assembler for Intel's iAPX 86 and iAPX 286 processors. Previously developed applications using any of these languages port easily to iRMX-based systems with minimal source code modifications.

In addition to the wealth of languages available, iRMX-based systems are complemented by utilities with which to create and manage object modules. This latitude in configurability allows programmers to team their efforts in order to achieve a shorter development time than would otherwise be possible.

Because the high-level languages are actually resident on the iRMX-based system, OEMs can pass application software directly on to end users. End users may then tailor the OEM's system to better meet application needs by writing programs using the same languages.

Language-Independent Application Development

Intel's Universal Development Interface (UDI) and Object Module Format (OMF) enable several users to write different modules of an application, in different languages, then link them together.

The OMF provides users with the ability to mix languages on a single application system, affording the luxury of choosing exactly the right language tools for specific pieces of the application, rather than compromising specialized tasks for the sake of one, project-wide language.

iRMX languages are fully compatible with the Intel Series III and Series IV Development Systems, should the user choose to develop applications on a specialized development system. Applications are easily moved to the final target system for test, debug and minor redevelopment.

Fast, Lean Programs for Rapid Processing

The iRMX language products enable programmers to write the smallest, fastest programs available in high-level languages, due to the compiler's superior ability to optimize code.

It is also possible to make iRMX operating system calls directly from FORTRAN, PASCAL and PL/M. This means that application developers can take full advantage of the iRMX multi-tasking capability, whereby multiple applications execute concurrently on the operating system. Multi-tasking, a requirement of most real-time systems, is sometimes as necessary in application software development as in an operating system environment.

Standardized REALMATH Support

All the iRMX languages (except BASIC and C) support the REALMATH floating point standard. This ensures universal consistency in numeric computation results and enables the user to take advantage of the Intel iAPX 86/20 and iAPX 88/20 Numeric Data Processor or iSBC® 337 MULTIMODULE™ boards, which boost performance two to four times over that possible on a mini-computer.

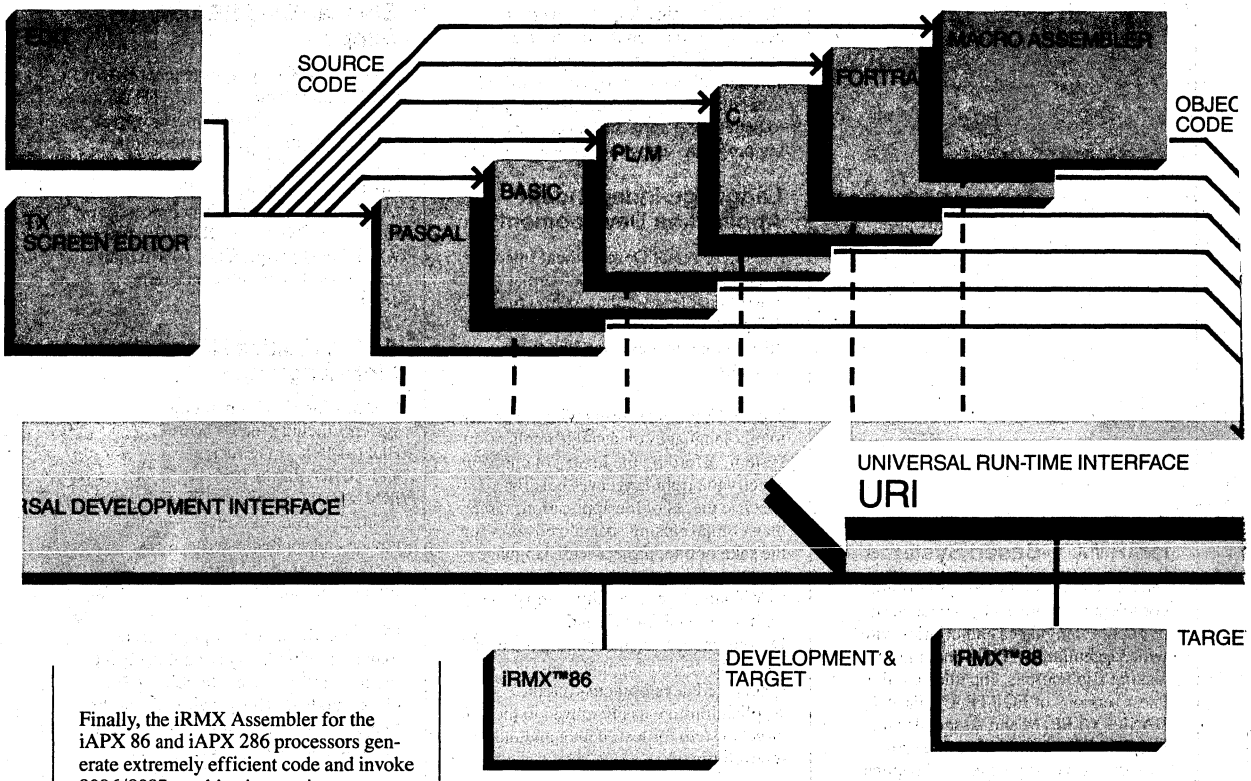
All the Utilities Needed to Link Languages

Utilities for iRMX operating systems include Intel's own EDIT, LINK, LOCATE and LIBRARIAN. The iRMX EDIT program meets the needs of both novice and sophisticated users with powerful line-oriented editing facilities.

Using the iRMX LINK program, users may link individually compiled object modules to form a single, relocatable object module. This provides the ability to merge work from several programmers into one cohesive application system.

The iRMX LOCATE utility maps relocatable object code into the processor memory segments, allowing user definition of module/memory type allocation. For example, often-used portions of an application may be mapped to (P)ROM.

The LIBRARIAN object code library manager affords easy creation, collection and maintenance of related object code to reduce the overhead of separately maintained modules.



Finally, the iRMX Assembler for the iAPX 86 and iAPX 286 processors generate extremely efficient code and invoke 8086/8087 machine instructions.

iRMX™ 86 Pascal

iRMX Pascal meets the proposed ISO language standard and implements several microcomputer extensions. A compile-time option checks conformance to the standard, making it easy to write uniform code. Industry-standard specifications contribute to portability of application programs and provide greater reliability.

iRMX 86 Pascal supports extensions, such as an interrupt-handler and direct port I/O extension, that allow programs to be written specifically for microcomputers. Separate module compilation allows linkage of Pascal modules with modules written in other high-level languages.

For more information on iRMX 86 Pascal see the Pascal 86 Software Package data sheet.

iRMX™ 86 FORTRAN

The iRMX 86 FORTRAN compiler provides total compatibility with FORTRAN 66 language standards, plus most new features provided by the FORTRAN 77 language standard. iRMX 86 FORTRAN includes extensions specifically for microcomputer application development. Programming is simplified by relocatable object libraries, which provide run-time support for execution time activities.

iRMX 86 FORTRAN supports the 8087 math coprocessor for the most powerful

microcomputer solution available in number-intensive applications. For more information on iRMX 86 FORTRAN see the FORTRAN 86 Software Package data sheet.

iRMX™ 86 PL/M

PL/M offers full access to micro-computer architecture while simultaneously offering all the benefits of a high-level language. Invented by Intel in 1976, PL/M 80 was the first microcomputer-specific, block-structured, high-level language available. Since then, thousands of users have generated code for millions of microcomputer-based systems using PL/M 80 and PL/M 86.

Software written for 8-bit processors (PL/M 80) are easily ported to the more powerful 16-bit (PL/M 86) environment. The same portability will be available for future VLSI.

For more information about iRMX 86 PL/M see the PL/M 86/88 Software Package data sheet.

iRMX™ 86 BASIC

Intel's offering of Microsoft BASIC is a standardized version of the most popular high-level language in the world. Existing BASIC programs are easily ported to iRMX-based systems. BASIC is an excellent pass-through language by which an OEM can offer customers the ability to write and modify their own applications.

iRMX™ 86 C Compiler

The popular new programming language, C (Mark William's Company version), is fully supported on iRMX-based systems. iRMX 86 C offers both small and large segmentation models, enabling applications to be written efficiently. The iRMX 86 C compiler combines assembly language efficiency with high-level language convenience; it can manipulate on a machine-address level while maintaining the power and speed of a structured language.

The iRMX 86 C compiler affords easy portability of existing C programs to iRMX-based systems. For more information on the iRMX C compiler see the iRMX 86 C Software Package data sheet.

iRMX™ 86 Text Editor

The iRMX 86 Text Editor is screen-oriented, menu-driven and easy to learn. Guided by the menu of commands always before him, the user can edit text and programs easily and efficiently.

iRMX 86 Text Editor allows the simultaneous edit of two files. This allows easy transferral of text between files and use of existing material in the creation of new files. Creating macros, strings of frequently-used commands, is also very simple. The editor "remembers" the selected commands and allows the user to re-use them repeatedly.

Worldwide Service and Support

All iRMX systems are completely supported by Intel's worldwide staff of trained hardware and software engineers. iRMX Language customers receive a warranty that includes Hotline Support, Software Updates, and Subscription Service.

Complete documentation is provided for all operating system and application software languages, as well as for system hardware components. An Intel system is not a collection of hardware and software pieces as much as a cohesive whole that is supported and serviced as such.

Intel Has Total Solutions for Real-Time Systems

iRMX 86 is the fastest, most powerful operating system available for multi-tasking, multi-user, real-time applications. Complemented by a wide range of industry-standard languages and utilities, the iRMX-based systems are highly flexible and configurable.

Application development for iRMX-based systems is possible at the board or the system level. OEMs can integrate functionality at the most profitable level of product design, using one system for both development and target use. Intel's choice of industry standard high-level languages enables the end user to extend OEM-provided functionality even further, if desired.

Who is better qualified to write and supply software for Intel VLSI than Intel? Today you have the ability to tap into hundreds of available application software packages, languages and utilities, peripherals and controllers and MULTIBUS® boards.

Tomorrow, and ten years down the road, you will be able to tap into the latest, high-performance VLSI—without losing today's software investment.

Specifications

Required Hardware

- Any iAPX 86/286 based or iSBC 86/286 based system including Intel's System 86/300 and 286/300 family. In addition, object code from the compilers will run on iAPX 88 based systems.
- 140KB of memory
- Two iRMX 86 compatible floppy disks or one hard disk
- One 8" double density or 5.25" double-density floppy disk drive for distribution of software
- System console device

Required Software

The iRMX 86 Operating System Release 6 or later including the nucleus, basic I/O system, extended I/O system and human interface

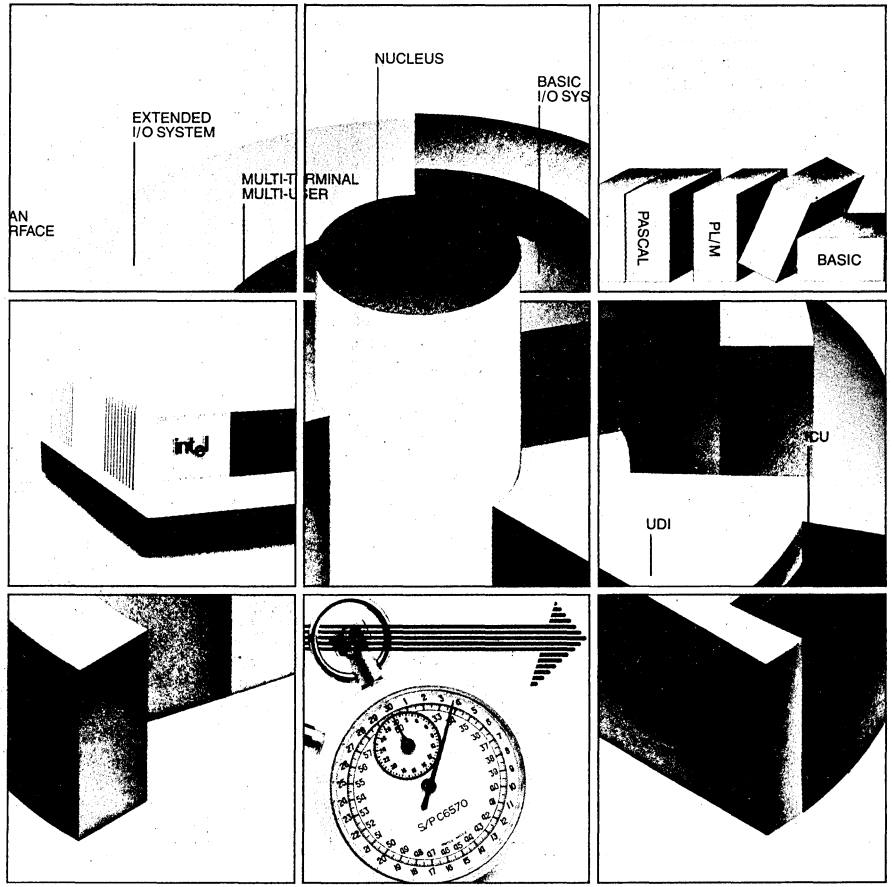
Purchase of any RMX language requires signing of Intel's OEM License Agreement (OLA).

Ordering Information

Language	Order Code	Product Contents	Warranty
ASM 86, Utilities	RMX 860	Two 8" disk and two 5.25" diskettes Edit Reference Manual iAPX 86/88 Family Utilities User's Guide Macro Assembler Operating Instructions ASM 86 Language Reference Manual 8087 Support Library Reference Manual	**90 days:** Software Updates, Subscription Service, Hotline Support
Pascal	RMX 861	Two 8" diskettes and two 5.25" diskettes Pascal 86 User's Guide	**90 days:** Software Updates, Subscription Service, Hotline Support
FORTRAN	RMX 862	Two 8" diskettes and two 5.25" diskettes FORTRAN 86 User's Guide	**90 days:** Software Updates, Subscription Service, Hotline Support
PL/M	RMX 863	One 8" diskette and one 5.25" diskette PL/M 86 User's Guide	**90 days:** Software Updates, Subscription Service, Hotline Support
TX Editor	RMX 864	One 8" diskette and one 5.25" diskette TX Screen Echter User's Guide	**90 days:** Software Updates, Subscription Service
BASIC	RMX 865	One 8" diskette and one 5.25" diskette BASIC Reference Manual BASIC 86 User's Guide One 8" diskette and one 5.25" diskette	**90 days:** Software Updates, Subscription Service
C	RMX 866	One 8" diskette and one 5.25" diskette C *Programming Language* by Kernighan and Ritchie (Prentice-Hall) C 86 Compiler User's Guide	**90 days:** Software Updates, Subscription Service, Hotline Support

iRMX™ OPERATING SYSTEM

- **High-performance, real-time, multi-tasking operating system for Intel's 86/300 and 286/300 microcomputer systems**
- **Highly configurable, modular structure for easy system expansion**
- **Wealth of design facilities and industry-standard languages to support fast, easy development**
- **Application software portable to next generation of Intel VLSI**
- **Supported by Intel's post-sales software support organization**



ORDER NUMBER: 230751-003

The Total Solution for the Real-Time Application OEM

Intel's iRMX™ 86 Operating System is a real-time, multi-tasking, multiuser, multiprogramming operating system designed to support high performance, time-critical applications such as factory automation, industrial control and communications networks. The iRMX operating system serves as an optimized event-driven executive for managing and extending the resources of Intel's 86/300 and 286/300 systems in real-time applications where high speed and low interrupt latency are required. Added performance for demanding numeric-intensive tasks comes from support of Intel's floating point math coprocessors.

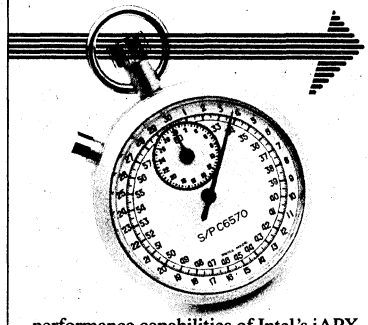
Comprised of modular layers, Intel's iRMX operating system is highly configurable, allowing the OEM to easily customize the system to meet the needs of target applications. In addition, the iRMX operating system provides OEMs with complete development capabilities. It has systems debuggers, crash analyzers, screen editors, utilities, and an Interactive Configuration Utility (ICU)—everything the development engineer needs to design and configure efficiently.

To further reduce development time, a complete set of industry-standard languages enables OEMs to take advantage of existing application software. This shaves months off development time and is a key advantage to the competitive OEM.

Speed, the Name of the Real-Time Game

In a real-time system the computer must respond to interrupts instantly; time is always at a premium. Intel's iRMX Operating System delivers superior real-time performance, thanks to ultra-fast context switching, task synchronization and memory-based message passing.

The iRMX 86 Operating System manages the resources of the 286/300 systems in real-address mode. iRMX 86 makes possible the utilization of the high-



performance capabilities of Intel's iAPX 286 microprocessor for those demanding high-speed applications.

Further accelerating processing power in number-crunching and floating point math applications is iRMX operating system's support of Intel's math coprocessors.

Our 8087 numeric data processor in our iRMX 86-based systems can perform floating point operations four times faster than competitive minicomputers with hardware math processors. For even greater performance, OEMs can select the iAPX 286 and the 80287 coprocessor working in tandem in the iRMX 86 system.

The superior price/performance ratio that results from combining Intel's iRMX operating systems and the System 300 family makes the choice clear: a more competitive Intel micro-based system over a more expensive minicomputer-based system.

Add More Processors for More Power, More Speed

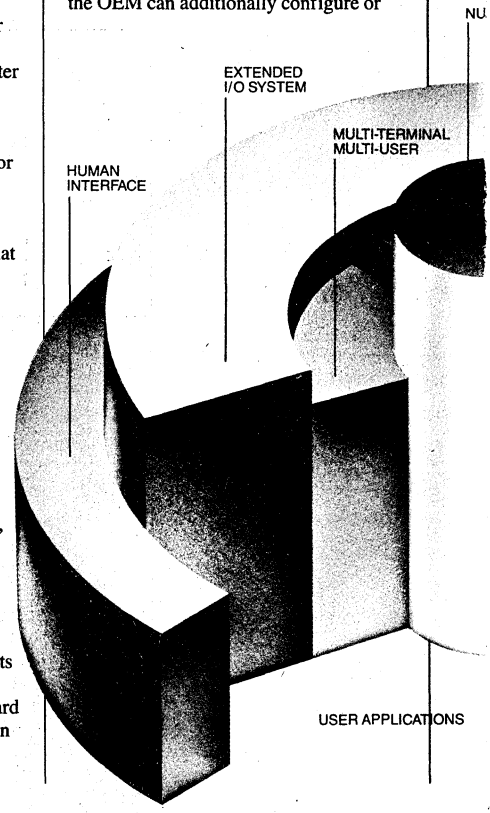
Need still more micro-muscle in your application? In an iRMX-based system, additional intelligent boards can be added to enhance system throughput.

With the iMMX™ 800 (MULTIBUS® Message Exchange) software package, the iRMX 86 Operating System supports a loosely-coupled multiprocessing environment. Tasks running on one board may communicate with tasks running on other boards, even if they operate under different Intel operating systems or microprocessors.

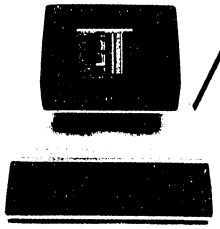
Multiprocessing is possible due to the hardware capabilities of Intel's System 300 MULTIBUS System Bus and the software support provided by iMMX™ 800. Overall system performance and flexibility can be greatly enhanced by off-loading the main CPU with such intelligent I/O boards as Intel's quad serial communication controller, digital controller or Ethernet communications controller.

Modular Software for Versatile, Easy Configuration

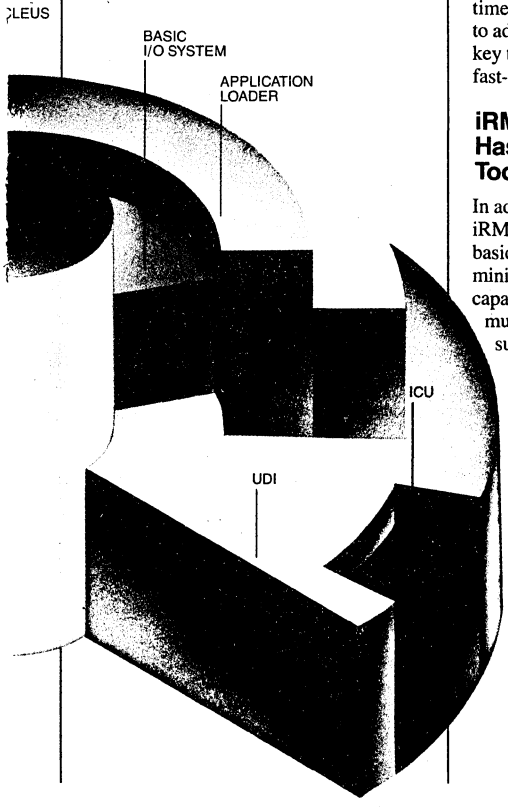
The iRMX operating systems shipped with Intel's 86/300 and 286/300 hardware systems are preconfigured at the factory to support a standard board set; however, the OEM can additionally configure or



extend the operating system to meet specific needs.



Intel's iRMX operating systems are configurable by system layer and by system call within each layer. Such flexibility gives designers the ability to choose software features that best suit their application's size and functional requirements. The iRMX Operating System also includes I/O drivers for many of Intel's MULTIBUS boards and industry-standard peripherals. You simply select the ones you need.



The Interactive Configuration Utility (ICU) is a built-in facility for assisting the OEM in the configuration process. The ICU prompts the user for system parameters and requirements, then builds a command file to compile, assemble, link, and locate necessary files.

The net results for the OEM: fast, easy system configuration with quick time-to-market benefits.

For customizing and extending your iRMX system, Intel has provided all the "hooks" necessary to make the job easy. The iRMX 86 Operating System contains extendability features that enable the OEM to add custom operating system calls, custom features, and custom functionality to his application—at any time in the application's life. The ability to add functions late in a product's life is key to an OEM's competitive edge in a fast-changing market.

iRMX™ Operating System Has All the Fundamentals, Too!

In addition to multiprocessing, Intel's iRMX operating systems have all the basics you would expect to find in a minicomputer operating system... capabilities such as multitasking, multiprogramming, and multiterminal support.

Multitasking requires a method of managing the different processes of an application and for allowing these processes to communicate with each other. The iRMX Nucleus provides these facilities plus task scheduling. The Basic I/O System provides users with the system calls for direct management of I/O devices needed for real-time applications. The Extended I/O System adds a number of I/O management capabilities to simplify access to files, such as automatic buffering and synchronization of I/O requests.

The Human Interface functions give users and applications simple access to the file and system management capabilities. Using the multiterminal support provided by the Basic I/O system, the Human Interface can support several simultaneous users. For example, multi-terminal support allows one person to use the iRMX Editor, while another compiles a FORTRAN or Pascal program, while several others load and access applications.

On-Target Development: One System Does It All

The beauty of Intel systems lies in their flexibility. Engineers developing an iRMX-based target system can use the same iRMX-based system in the development process; the development and target systems are one in the same. The bottom-line benefit is low entry-level costs for the OEM.

On-target development contributes immeasurably to a shorter development curve and decreased time-to-market, since it isn't necessary to purchase and learn separate development systems. With Intel's iRMX-based system, one system does it all.

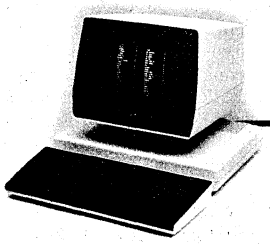
Tap into a Wide Range of Languages and Utilities

An Intel iRMX-based system supports many industry-standard and widely available languages: FORTRAN 77, Pascal (ISO Draft Standard) and PL/M compilers; Intel Assemblers, and popular independent vendor products, such as Microsoft's BASIC and Mark Williams' C compiler.

iRMX operating systems also have a menu-driven, screen-oriented text editor and a variety of utilities for manipulating

object code to facilitate the development process.

Multiple-language support is made possible by a set of systems calls known



as the Universal Development Interface (UDI) which enables the iRMX systems to interface with many compilers and language translators. UDI ensures that users will be able to transport applications to future releases of iRMX operating systems as well as use language and utilities of other software vendors that support UDI. (For more information on Intel iRMX languages, see the iRMX Language Fact Sheet)

As an option, a commercial extension package iCEX is available. It provides such useful utilities as: a Shared I/O System (SIOS) that allows multiple tasks to access mass storage data through shared buffers in main storage; a Re-entrant Program Manager (RPM) that eliminates the need to have multiple copies of the same program in memory to support concurrent applications; a File Printer; Multi-user LOG ON facilities; and many more.

Intel's Open Systems Approach Means Freedom to Grow

At Intel, we believe that systems need to expand in order to meet the needs of a changing market; and that is how we design our products.

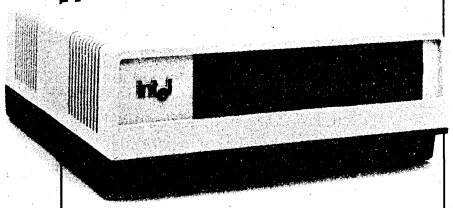
Standards are the key to systems that are open to future expansion, future technology and future markets.

Intel's iRMX operating systems are built from the inside-out with industry standards: UDI (Universal Development Interface), RTI (Runtime Interface), MULTIBUS System Bus (IEEE 796), iMMX 800 Package (MULTIBUS multi processing), Ethernet (IEEE 802.3), extended math format (IEEE P754), and industry-standard peripheral device interfaces.

An OEM who builds his product around one of Intel's RMX-board systems is assured of multi-vendor hardware/software alternatives and a future upgrade path. In today's highly competitive markets, that is the only kind of system to build.

Today, you'll have the ability to tap into readily available application software packages, languages, and utilities, MULTIBUS boards, and peripherals. Tomorrow, you will be able to tap into the latest, high-performance VLSI without sacrificing today's software investment. Applications written on iRMX 86 will run on Intel's iAPX 86, iAPX 88, iAPX 186, iAPX 188 and iAPX 286-based systems.

Not to be forgotten are the advantages of starting from the systems level to begin with. Intel has invested hundreds of man-years in software and hardware development for its systems products. For the OEM trying to meet a market window, time-to-market is much faster when starting with a system instead of boards or components. It makes good business sense to let Intel provide the "micro-engine", so you can concentrate on your area of expertise and get to market sooner!



Worldwide Service and Support

The iRMX 86 Operating System is a mature proven product with thousands of installations at the component, board and systems levels. Post-sales software support is available to Intel iRMX 86 Operating System OEMs in the form of software updates and routine systems software maintenance. Software support is extendable in one-year increments after the initial 90-day warranty. Hotline service is available separately to customers needing quick regional software support. All software is completely documented, and users receive monthly technical reports, newsletters and access to the iRMX users group and software libraries.

iRMX users can also take advantage of Intel's worldwide staff of trained hardware and software engineers for application design assistance. We offer complete training for operating system software and associated system hardware, bringing OEM's up to speed and helping get their products to market quickly.

Intel, the Technology Leader ...With the Total Solution

Intel started the microprocessor revolution with the 4004 and has been the market leader with every generation of advanced microprocessor VLSI since. We not only invented the microprocessor but MULTIBUS single board computers, as well.

Intel's technology leadership has, by necessity, extended from microprocessors into operating system software. iRMX is recognized as the industry standard real-time VLSI operating system.

OEMs can enhance their product's marketability by leveraging their value-added on top of the solid foundation of an iRMX-based Intel 300 microcomputer system. Intel's solution offers the most price/performance with the least risk to progressive OEMs... because we know the real-time game from the inside out.

RMX™ OPERATING SYSTEM

Specifications

Supported Software Products

iRMX 860	iRMX 86 Development Utilities Package including the iAPX 86 and 88 Linker, Locator, Macro Assembler, Librarian, and the iRMX 86 Editor
iRMX 861	Pascal 86/88 Compiler
iRMX 862	FORTRAN 86/88 Compiler
iRMX 863	PL/M 86/88 Compiler
iRMX 864	TX-Screen-Oriented Editor
iRMX 865	BASIC Interpreter
iRMX 866	C Compiler
iMMX 800	MULTIBUS® Message Exchange software package for iRMX 80, 86, 88, and 286 application systems

Supported Hardware Products

iSBC® MULTIBUS® Products

iSBC 86/12A, 86/05, 86/14, 86/30, 86/35, 88/25, 88/40, and 286/10 Single Board Computers

iSBC 186/03 Single Board Computer

iSBC 186/51 Ethernet Controller

iSBC 188/48 Communications Controller

iSBC 286/10 Single Board Computer (Real Address Mode only)

iSBC 204	Flexible Disk Controller
iSBC 206	Hard Disk Controller
iSBC 208	Flexible Disk Controller
iSBC 215	Winchester Disk Controller
iSBC 220	SMD Disk Controller
iSBX 251	Bubble Memory System
iSBC 254	Bubble Memory System
iSBC 534	4-Channel Terminal Interface
iSBC 544	Intelligent 4-Channel Terminal Interface and Controller
iSBX 218	Flexible Disk Controller
iSBX 350	Parallel Port (Centronix-type Printer Interface)
iSBX 351	Serial Communications Port
iSBX 270	CRT, Light Pen and Keyboard Interface

System 86/300 Family

System 286/300 Family

Available Literature

The iRMX 86 Documentation Set is comprised of the following four volumes of reference manuals. Order numbers are associated with these four volumes only.

iRMX 86 Introduction and Operator's Reference Manual for Release 6
Order Number: 146545-001

Introduction to the iRMX 86 Operating System

iRMX 86 Operator's Manual

iRMX 86 Disk Verification Utility Reference Manual

iRMX 86 Programmers Reference Manual for Release 6, Part 1
Order Number: 146546-001

iRMX 86 Nucleus Reference Manual

iRMX 86 Basic I/O System Reference Manual

iRMX 86 Extended I/O System Reference Manual

iRMX 86 Programmer's Reference Manual for Release 6, Part II
Order Number 146547-001

iRMX 86 Application Loader Reference Manual

iRMX 86 Human Interface Reference Manual

iRMX 86 Universal Development Interface Reference Manual

Guide to Writing Device Drivers for iRMX 86 and iRMX 88 I/O Systems

iRMX 86 Programming Techniques

iRMX 86 Terminal Handler Reference Manual

iRMX 86 Debugger Reference Manual

iRMX 86 System Debugger Reference Manual

iRMX 86 Crash Analyzer Reference Manual

iRMX 86 Bootstrap Loader Reference Manual

iRMX 86 Installation and Configuration Guide for Release 6
Order Number: 146548-001

iRMX 86 Installation Guide

iRMX 86 Configuration Guide

Master Index for Release 6 of the iRMX 86 Operating System

iRMX™ 86 Configuration Size Chart

System Layer	Min. ROMable Size	Max. Size	Data Size
Bootstrap Loader	1K	1.5K	6K*
Nucleus	10.5K	24K	2K
BIOS	26K	78K	1K
Application Loader	4K	10K	2K
EIOS	10.5K	12.5K	1K
Human Interface	22K	22K	15K
UDI	11K	11K	0
Terminal Handler	3K	3K	0.3K
Debugger	28.5K	28.5K	1K
Human Interface Commands			116K
Interactive Configuration Utility			308K

System 86/300 Memory:	348KB
Maximum Addressable Memory:	1MB
Minimum Memory Required with ICU Loaded:	448KB

*Usable by System after Bootloading.

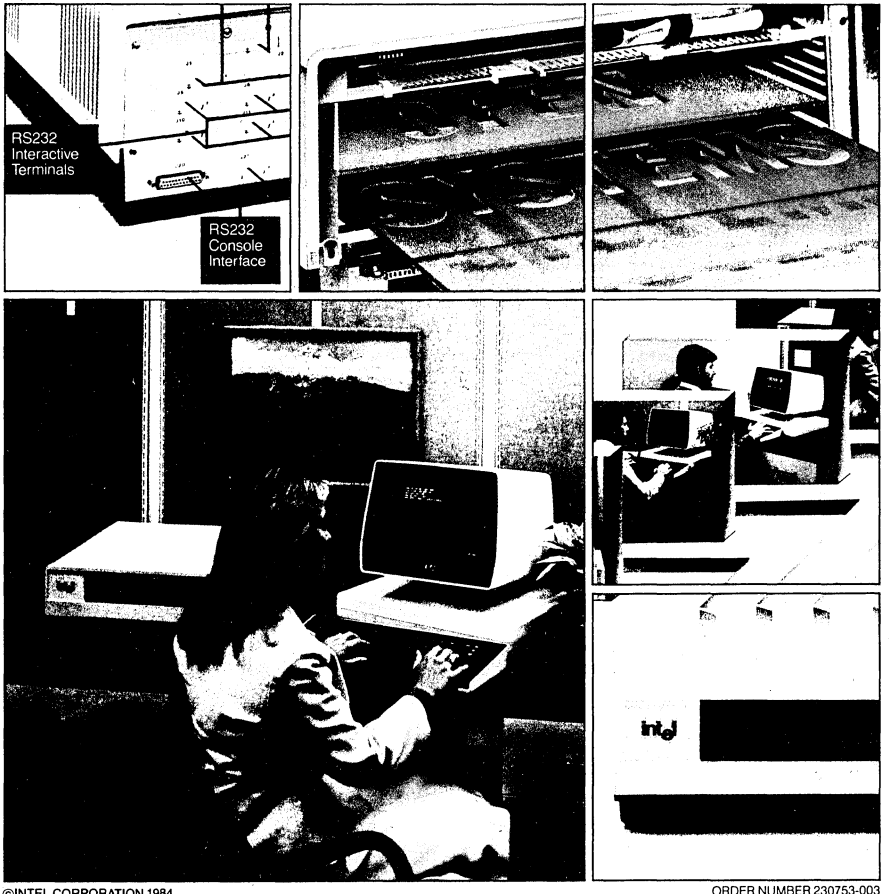
Ordering Information

Each iRMX operating system includes two startup systems supporting Intel's System 300 standard hardware and Intel processor boards. Intel System customers also receive the iRMX 860 (Assembler, Linker, Locator, Libraries, Editor, Utilities) and iRMX 863 (PL/M Language) products and are entitled to one prepaid incorporation fee. Also included: Software Problem Reporting Service (SPR), and a 90 day System Software Subscription (new s/w release updates). Also includes System Software documentation.

Refer to Intel's OEM price list, OEM Microcomputer System section, for ordering information.

SYSTEM 310 MICROCOMPUTER SYSTEM

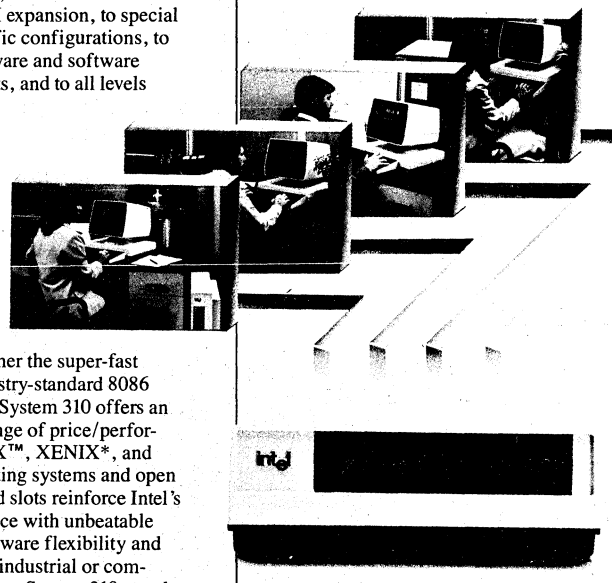
- **High-performance, highly expandable OEM microcomputer system based on Intel's 80286/80287 supermicro and 8086/8087 world-standard processors**
- **Open MULTIBUS® architecture for flexibility and future growth**
- **5¼" Winchester disks and diskette drives for data storage, backup, and interchange**
- **Industry-standard operating systems and programming languages**
- **Total system support from Intel's world-wide field service and support organization**



ORDER NUMBER 230753-003

High-performance, Expandable Microcomputer System for the OEM

Intel's System 310 microcomputer offers OEMs an open, MULTIBUS-based 16-bit microcomputer system for high-speed real-time applications and interactive multi-user commercial and technical environments. Everything about System 310 is standard—buses, interfaces, peripherals, and software—and open to new VLSI expansion, to special application-specific configurations, to aftermarket hardware and software suppliers' products, and to all levels of integration.



Available with either the super-fast 80286 or the industry-standard 8086 microprocessors, System 310 offers an unprecedented range of price/performance. The iRMX™, XENIX*, and MSDOS** operating systems and open MULTIBUS board slots reinforce Intel's system performance with unbeatable software and hardware flexibility and expansion for the industrial or commercial OEM. Also, System 310 may be placed on a desktop, rackmounted, or installed in an optional floorstand.

Finally, everything in the System 310 is supported by Intel: boards, chassis, software and peripherals are purchased and serviced through one-stop shopping. When you buy a system from Intel, we don't send you to one outside vendor for software support and to another for hardware repairs. One phone call is all you need.

Two "Micro-Engines" to Choose From

At the heart of the 310 systems, Systems 286/310 and 86/310, are the industry's leading microprocessors: the Intel 80286 and 8086, respectively. The OEM can select the optimal price/performance hardware mix for each application.

System 286/310 is based on the Intel 80286, the outstanding supermicro on the OEM market today. Residing on an Intel single board computer, the 80286 delivers more than twice the performance of the 8086 while its teammate, the 80287 numeric coprocessor, assists in providing System 286/310 with unmatched throughput in its price class for numeric-based applications.

Intel's iLBX™ (Local Bus Exchange) on System 286/310 turns the processor and memory boards into a "virtual single board" for the fastest possible data and instruction access in high-performance applications, and the OEM can expand with up to two additional memory boards that exploit the iLBX bus's fast-access characteristics.

System 86/310, based on the industry-standard 8086, offers new low cost levels for 16-bit integrated systems while leaving the path open to move 86-based applications up to higher performance on System 286/310. System compatibility maintained at the operating system level protects your software investment as your needs change.

Ample, Flexible Storage for Unlimited Application Growth

Industry-standard 5¼ inch peripheral drives furnish data and program storage: a 320KB diskette drive in all systems, and depending on the configuration, no Winchester hard disk drive, a 12MB, 19MB or 40MB Winchester hard disk drive. Increases in available disk storage will keep pace with advances in Winchester and other peripheral technologies; System 310-compatible peripheral chassis also offers tape and multiple Winchester drive configurations. All systems ship with high-speed RAM memory: 1MB ECC in the largest System 286/310, 640KB in the largest System 86/310 with the ability to increase memory to a total of 4.0MB (286-based) and 896KB (86-based), respectively. The variety and quantity of storage media available for System 310 means that the OEM will never have to curb application growth due to lack of memory.

Highly Expandable to Meet Diverse OEM Application Needs

Intel's System 310 offers the OEM system expansion and flexibility because it is MULTIBUS-based. Intel's industry leadership in standardizing the bus (IEEE 796) is supported by 170 vendors with over 1,000 currently-available products.

System 310 configurations offer between three and six MULTIBUS board slots for expansion and customization. The expansion possibilities are almost limitless: disk controllers, communications boards, graphics interfaces, memory expansion boards, and many other specialized single board computer products are available to the OEM con-

*XENIX is a trademark of Microsoft Corporation.
**MSDOS is a trademark of Microsoft Corporation.

figuring unique functionality into his product.

If the time comes that a System 310 configuration needs even more power, the OEM can take advantage of MULTIBUS' multiprocessing architecture. Multiple processors can share computing and I/O loads, with standard system software in support of real-time communication between boards. Multiprocessing is a standard feature of MULTIBUS when one processor isn't enough.

Building end-user systems on an industry-standard foundation such as MULTIBUS means much more than having a smorgasbord of available third-party options. It is a guarantee of continued system compatibility and expandability for years to come. OEMs can absorb state-of-the-art technology as quickly as it becomes available.

System Extension Modules for Easy I/O Expansion

For further expansion Intel provides System Extension Modules (iSXM™). iSXMs are factory-configured board, peripheral, and accessory products designed for installation in Intel System products to add I/O capability. For example, the iSXM 544 Intelligent Communications board supports four RS232 channels for multiterminal operations with iRMX or XENIX, plus it furnishes a parallel I/O interface compatible with the Bell 801 Automatic Calling Unit for auto-dial functions. Firmware coupling the iSXM 544 board with both operating systems is resident on the board as delivered. Other iSXM products provide low-cost or very-high-performance communications interfaces, I/O cable sets, and other expansion products.

Worldwide Compatibility

The System 310 is designed to be a "good citizen" throughout the world, operating on commercially-available power sources and adhering to a broad set of standards laid down by many nations for safety and signal radiation. The system meets UL and CSA safety requirements, as well as FCC standards for radio-frequency emissions (RFI) and electromagnetic interference (EMI). Additionally, it is designed to allow the OEM to be confident that he can meet the requirements of IEC 435 and VDE 0806 for safety and VDE 0871 for RFI/EMI.

Industry-standard Operating Systems and Languages Increase the OEM's Options

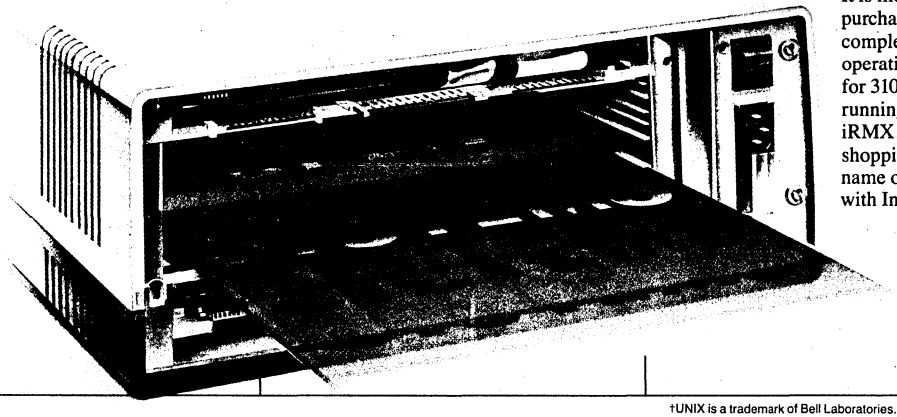
Intel's Open Systems standards philosophy extends to the System 310's software. Three industry-standard operating systems, iRMX, XENIX, and MSDOS allow the OEM to optimize systems for real-time or interactive commercial applications, and take advantage of many off-the-shelf MSDOS applications and languages. A wide range of popular high-level languages enable application developers to program in multiple languages and still link modules together.

Intel's XENIX operating system is Bell Laboratories' Version 7 UNIX† with a blend of enhancements from Microsoft and Intel to specifically suit OEM interactive, multi-user applications. Benchmark figures show Intel's XENIX 286 to be at the leading edge of UNIX performance on a microsystem. For application areas such as distributed data processing, business data processing, and software development, XENIX offers superior human-machine interaction and performance.

The iRMX operating system optimizes System 310 products for real-time, multitasking, time-critical applications such as factory automation, industrial control, and communications networks where rapid response to the "real world" is required.

Both operating systems are supported by a range of high-level languages for rapid and easy applications development. FORTRAN, COBOL, C, Pascal, PL/M, Intel Assemblers and popular utilities round out System 310's software offerings, providing the OEM with a comprehensive development tool kit for getting applications to market before the competition. In addition, a wide range of independent software vendors (ISV) are available on the System 310, both under iRMX and XENIX.

MSDOS is a complete and fully licensed version of Microsoft's MSDOS 2.11 operating system. It is most often purchased as a complementary operating system for 310 systems running XENIX or iRMX. One-stop shopping is the name of the game with Intel systems:



†UNIX is a trademark of Bell Laboratories.

hardware, software and all the necessary support are available from a single, reliable vendor.

Full System Support

Quality. Since Intel's systems are built from Intel boards and components, quality is checked and double-checked all along the way. All systems are run in to eliminate potential problems with "infant mortality"; our philosophy is that if a system component is going to break down, we want it to break down on us.

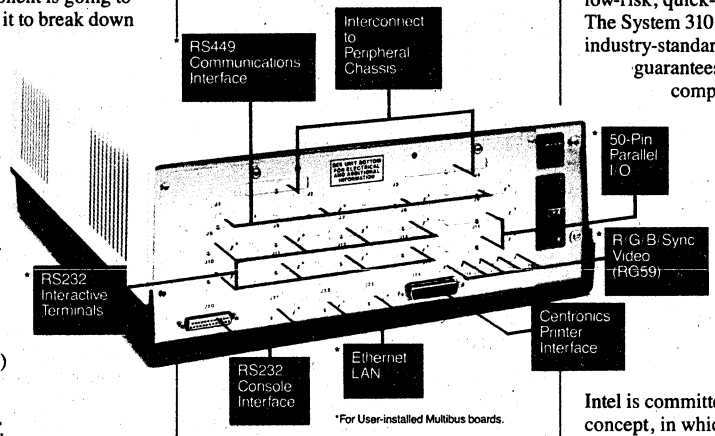
Every System 310 contains a comprehensive, two-part diagnostic package to assist in problem isolation and to further ensure system reliability. A System Confidence Test (SCT) automatically checks out the boards on power-up; a System Diagnostic Test (SDT) provides detailed diagnostics on bugs identified by the SCT.

OEMs also receive the System Analysis Test (SAT) with their XENIX and iRMX software. This test tool stresses both hardware and software to isolate hardware/software interaction problems.

Service. All hardware is warranted for 90 days. Maintenance contracts are available, or customers can opt for service on a per-call basis. If on-site repair is not required, 48-hour factory repair is available on a limited basis, as well as our economical direct-return service. Additionally, "Family Plan" service, covering non-Intel portions of the system as well as Intel-supplied parts, is available on a negotiated basis.

Software service is offered by a factory-resident group with consulting support, response to problem reports, and a Hot Line for critical problems that just won't wait.

Application support. Intel's application engineering organization is one of the largest in the world. We have Field Application Engineers specializing in everything from system software to complex peripherals. These professionals are dedicated to supporting OEMs in the pre-sales development environment. A separate group, Systems Engineers, is dedicated to post-sales consulting support. Intel's built-in quality, our 75-plus worldwide field service locations, and staff of trained hardware and software applications engineers are major advantages for the OEM trying to get an application running and, once running, having the systems maintained.



VLSI Systems Leadership —Today and Tomorrow

Intel is the undisputed leader in microprocessor VLSI; we invented both the microprocessor and the industry-standard MULTIBUS, and we are the world's largest producer of MULTIBUS board-level products.

With the System 310 and other products, we're expanding our leadership into the OEM microcomputer systems market. It's not unfamiliar turf. Intel has been in the systems business for a number of years; we're the largest supplier of microprocessor development systems in the world!

We've also introduced an end-user-oriented data base information system (iDIS) that accesses mainframe-resident data bases.

Intel is uniquely able to offer VLSI technology leadership to OEMs at the low-risk, quick-to-market systems level. The System 310's openness and industry-standard MULTIBUS are guarantees of continued system compatibility and expandability for years to come. With Intel systems, there are no dead ends lurking in the OEM's product future; rather, he can now absorb new VLSI advances and functionality almost as quickly as they become available.

Intel is committed to the Open Systems concept, in which industry standards pave the way into future applications and future markets. For the OEM seeking product wins in today's fiercely competitive markets, Open Systems from Intel are the only way to build.

Specifications

Central Processor: System 286/310 models provide an 80286 General Purpose Processor; System 86/310s provide an 8086 General Purpose Processor. Instructions are 8, 16, or 32 bits in length, data are 8- or 16-bits long; numeric processing (with Numeric Co-Processors) is carried out in 80-bit-words.

Mass Storage: See Configuration Summary for standard configurations; all drives are 5¼-inch format; the Winchester controller provides complete ECC write/read checks for data integrity.

System Expansion: Three to six Multibus (IEEE 796) slots for customizing and expansion based on over 80 boards available from Intel and more than 900 offered by independent hardware vendors.

Environmental Specifications:
Operating: 10°C to 40°C (Winchester Only); to 35°C (With Diskette) (26°C maximum wet bulb temperature) (20% to 80% Relative Humidity, non-condensing)
Altitude: Sea Level to 8,000 feet

Regulatory Agency Specifications:
Meets: UL 114—Safety; CSA 22.2—Safety; FCC Docket 20780—RFI/EMI.
Designed To Meet: IEC 435—Safety; VDE 0871—RFI/EMI

AC Power Input: 88-132VAC or 180-264VAC, 47-63Hz (user-selectable on chassis)

310 Dimensions: Height: 6½", Width: 17", Depth: 22"

Weight: Less than 50 lbs. (Varies with configuration)

Configuration Summary

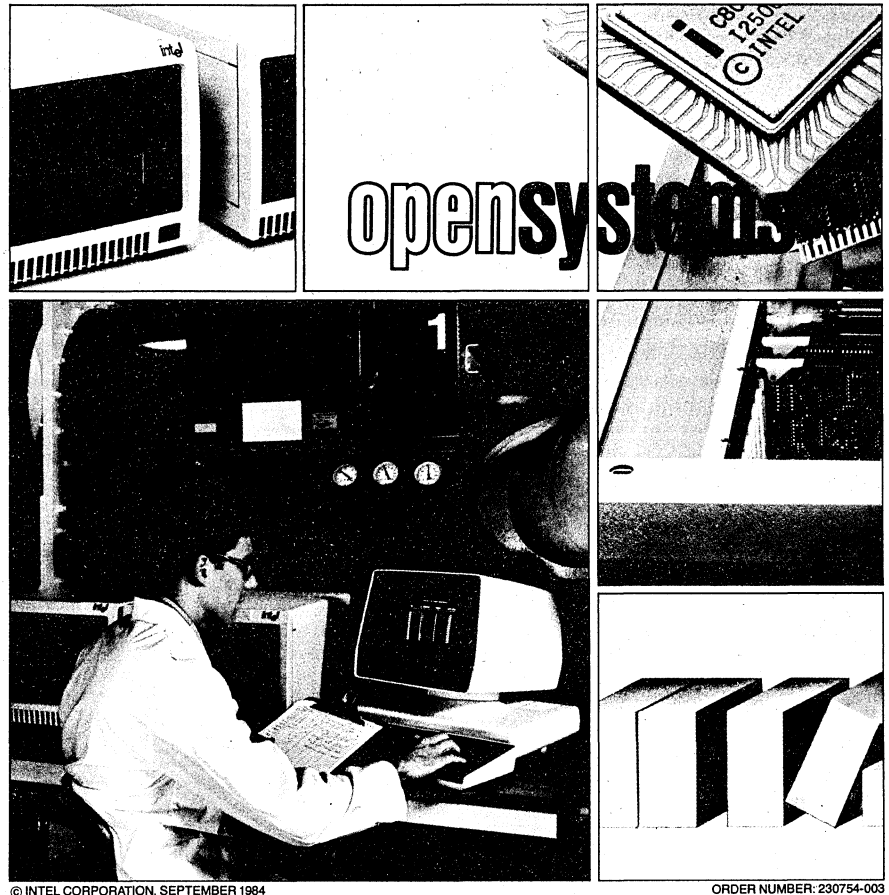
	System 286/310-41	System 286/310-40	System 286/310-17	System 286/310-4	System 86/310-3A, -3	System 86/310-2A, -2	System 86/310-1
Microprocessor	80286 (6 MHz)	80286 (6 MHz)	80286 (6 MHz)	80286 (6 MHz)	8086 (5 MHz)	8086 (5 MHz)	8086 (5 MHz)
Numeric Coprocessor	80287 (4 MHz)	80287 (4 MHz)	80287 (4 MHz)	80287 (4 MHz)	8087 (5 MHz)	8087 (5 MHz)	N/A
RAM Memory, Expandable To	1MB W/ECC 4.0MB W/ECC	1MB W/ECC 4.0MB W/ECC	512KB W/ECC 4.0MB W/ECC	512KB W/ECC 4.0MB W/ECC	640KB 896KB	256KB 896KB	128KB 896KB
Mass Storage	320KB Diskette (Formatted) 40MB Wini (Unformatted)	320KB Diskette (Formatted) 40MB Wini (Unformatted)	320KB Diskette (Formatted) 19MB Wini (Unformatted)	320KB Diskette (Formatted)	320KB Diskette (Formatted) 19MB Wini (12MB in **-3**) (Unformatted)	320KB Diskette (Formatted) 19MB Wini (12MB in **-2**) (Unformatted)	320KB Diskette (Formatted)
I/O Ports	(10) RS232 (1) Centronics	(2) RS232 (1) Centronics	(2) RS232 (1) Centronics	(2) RS232 (1) Centronics	(1) RS232 (1) Centronics	(1) RS232 (1) Centronics	(1) RS232 (1) Centronics
MULTIBUS® Expansion Slots	3 @ 0.65 in.	4 @ 0.65 in.	4 @ 0.65 in.	1 @ 1.20 in. 4 @ 0.65 in.	4 @ 0.65 in.	5 @ 0.65 in.	1 @ 1.20 in. 5 @ 0.65 in.
DC Power Output	270 Watts Maximum +5V @45A +12V @4.7A −12V @4.7A	270 Watts Maximum +5V @45A +12V @4.7A −12V @4.7A	220 Watts Maximum +5V @30A +12V @4.7A −12V @4.7A	220 Watts Maximum +5V @30A +12V @4.7A −12V @4.7A	220 Watts Maximum +5V @30A +12V @4.7A −12V @4.7A	220 Watts Maximum +5V @30A +12V @4.7A −12V @4.7A	220 Watts Maximum +5V @30A +12V @4.7A −12V @4.7A

Ordering Information—System Hardware

System Description	Order Code
System 286/310-41	SYS310-41A
System 286/310-40	SYS310-40A
System 286/310-17	SYS310-17A
System 286/310-4	SYS310-4A
System 86/310-3	SYP310-3
System 86/310-3A	SYP310-3A
System 86/310-2	SYP310-2
System 86/310-2A	SYP310-2A
System 86/310-1	SYP310-1

SYSTEM 380 MICROCOMPUTER SYSTEMS

- **High-performance, highly expandable OEM microcomputer systems, iAPX 86- or iAPX 286-based**
- **High-performance floating point math capabilities with 8087/80287 numeric coprocessors**
- **Unmatched system flexibility with eleven MULTIBUS® expansion slots, 8-inch peripheral slot, and 13 back-panel knock-out ports**
- **35MB Winchester disk and 1 MB flexible diskette for data storage and backup**
- **Total system support from Intel's worldwide field service and support organization**



ORDER NUMBER: 230754-003

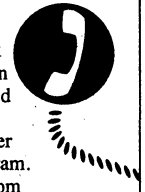
Highest Performance, Most Expandable Microcomputer System on the Market

Intel's System 380 microcomputer system is a MULTIBUS-based, integrated package targeted at OEM's needing a powerful, flexible, expandable base product upon which to add value.

Available in either iAPX 86 or a super-fast iAPX 286-based version, the System 380 offers high performance and expandability to the industrial or commercial OEM. With an industry-standard 35 MB Winchester, 1 MB diskette, eleven board slots and one peripheral slot for system expansion, the System 380 provides unmatched flexibility in a MULTIBUS system.

And, with optional iRMX or XENIX* operating systems and a broad array of popular languages and utilities, the System 380 meets the software needs of technical or commercial applications.

The System 380 is an open system; a system built around industry standards. This means that the OEM's product can easily incorporate the next generations of Very Large Scale Integration (VLSI) or take advantage of industry-standard hardware and application software for maximum configuration flexibility and quick time-to-market advantages. With today's fast-paced, rapidly changing technology, an open system just makes good business sense.

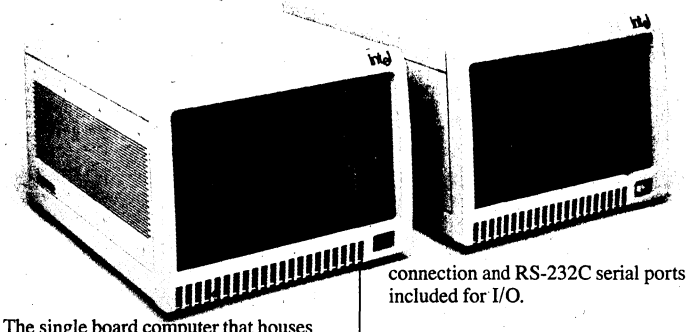


And to help reduce the risk of ownership, everything in the System 380 is supported by Intel: boards, software and peripherals are all under our one-stop support program. When you buy a system from Intel, you don't go to one outside vendor for software support and to another for hardware repairs. One phone call is all you need to keep your systems at peak-performance.

Two High-Performance "Micro-Engines" to Choose From

At the heart of System 86/380 and System 286/380 are the industry's leading microprocessors: the Intel iAPX 86 and iAPX 286, respectively. The OEM can select the optimum price/performance hardware mix that best fits the application.

System 86/380 is based on the Intel 8086, the industry's most popular 16-bit microprocessor.



The single board computer that houses the central processor also contains Intel's 8087 Numeric Data Processor. Working together, the 8086 and 8087 perform floating point processing at 50 times the throughput of an 8086 system alone and four times faster than competitive minicomputers with hardware math processors.

System 286/380 is based on the Intel 80286, the fastest 16-bit microprocessor available today. Residing on a single board computer, the 80286 is capable of delivering more than twice the performance of the 8086. When teamed with the 80287 math coprocessor, an 80286-based system is virtually unmatched in numeric-intensive applications. In addition, Intel's iLBX (Local Bus Extension) offers extremely fast memory access for high-performance applications.

Both systems contain a standard 35MB, 8-inch Winchester hard disk as well as a 1MB flexible disk drive for program and data storage. Both come with high-speed RAM memory (384 KB in the 86/380 and 512 KB ECC memory in the 286/380), expandable to more than ten megabytes. Systems are designed to meet UL/CSA/VDE safety regulations and FCC/VDE EMI/RFI requirements.

System 380 is available as a desktop or rack-mount dual-chassis package. The processor chassis contains processor, RAM, and disk controller boards, while the peripheral chassis contains the disk drive peripherals. One parallel printer connection and RS-232C serial ports are included for I/O.

The 380 systems are available as hardware only for the OEM's volume production needs or as hardware kitted with iRMX or XENIX system software with everything needed to get an application designed and into production.

The System 380's processing performance together with generous allocations of Winchester, flexible disk and RAM storage offer OEMs the resources to develop sophisticated industrial or commercial applications.

Highly Expandable to Meet Diverse OEM Applications

Intel's System 380 has more system expansion alternatives than any other microcomputer system on the market; Intel's MULTIBUS architecture is one of the reasons why. Intel's industry-standard MULTIBUS (IEEE 796) is supported by 150 vendors with over 1000 currently available products.

System 380 contains eleven MULTIBUS board slots for system expansion and

*XENIX is a trademark of Microsoft Corporation.

customization. The expansion possibilities for an Intel MULTIBUS-based system are nearly limitless: disk controllers, communications controllers, graphics controllers, bubble memory, memory expansion boards, and many other specialized MULTIBUS boards are available to the OEM configuring unique functionality into his product. And the MULTIBUS specification is widely published to encourage OEMs to build custom boards for their system.

In addition to adding MULTIBUS boards, the System 380's mass storage can be expanded by adding an eight-inch hard disk drive in the open peripheral slot. The additional peripheral, controlled by the system's internal controller board, can be added without an extra power supply; the existing one has plenty of power!

For flexibility and low-risk expandability, the System 380 can't be beat.

System Extension Modules for Easy I/O Expansion

For data communications expansion, Intel provides System Extension Modules (iSXMs). Intel's iSXMs are factory-configured MULTIBUS boards that plug directly into the System 380 expansion slots to allow the addition of more terminals and I/O devices.

For example, when installed in the System 380, the iSXM 534 module provides four additional fully programmable RS-232C serial communications channels, plus a parallel I/O interface that is compatible with the Bell 801 automatic calling unit (ACU).

The iSXM 544 module is an intelligent version of the iSXM 534 module acting as a communications controller or an intelligent slave for multi-terminal applications. The iSXM 953 module provides necessary cabling for four RS-232C serial ports. The iSXM 534 or iSXM 544 module plus an iSXM 953 module is the easiest way to add multiple terminals to a 380 system.

Industry-Standard Operating Systems and Languages Shorten Development Time

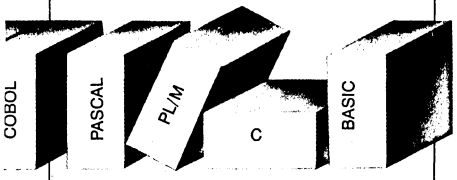
Intel's open systems, standards-only philosophy extends to the System 380's software. Two industry-standard operating systems, iRMX and XENIX, allow the OEM to adapt systems to real-time or interactive applications and a wide range of popular high-level languages enable easy development.

The iRMX operating system (available for both System 86/380 and the 286/380) is geared towards real-time, multitasking, time-critical applications such as factory automation, industrial control and communications networks where rapid response to machine-generated interrupts is required.

opensystems

Software developed on 86-based systems is totally upward portable to the 286-based systems.

Intel's XENIX operating system, also available for both systems, is Bell Laboratory's Version 7 UNIX* with a blend of enhancements from Microsoft and Intel



to suit interactive, multiuser applications. For application areas such as distributed data processing, business data processing and software development, XENIX offers superior user-machine interaction and performance.

Both operating systems are supported by a range of high-level languages for rapid and easy application development. FORTRAN, COBOL, Pascal, PL/M, "C", BASIC, Intel Assemblers and popular utilities round out the System 380's software offerings, providing the OEM with comprehensive development tools for getting applications to market fast. Additionally, Program Libraries and Users Groups link the entire community of system users together, leveraging each other's software investments.

Quality Systems with Support to Match

Intel's reputation for quality holds true at the systems level. Intel builds their systems from field-proven Intel MULTIBUS boards and operating systems.

Integrated peripherals and components must pass Intel's rigorous qualification standards before becoming part of an Intel system. Extensive life-cycle and environmental tests are performed during the development process to verify reliability. During production, functional operation is tested at the module and system levels for every unit. Extensive environmental testing is done periodically on product samples to maintain quality. Regular audits of customer and field service reports are used to continually monitor the reliability of field units. Intel understands how vital reliability is at the systems level, and we're committed to building it into our products.

Every System 380 also contains a comprehensive, tri-level diagnostic package to assist in problem isolation and assure system reliability. The System Confidence Test (SCT) automatically checks out all boards and peripherals on power-up; the System Diagnostic Test can isolate problems identified by the SCT to the board level; and the System Analysis Test stresses both hardware and software to isolate related problems.

All hardware is warranted for 90 days. Should repair service be required, the customer can choose on-site service under a maintenance agreement or on a per-call basis. If on-site repair is not required, 48-hour factory repair is available, in addition to our economical direct-return service.

Provided the necessary training and diagnostics, Intel will actually service non-Intel products such as terminals or printers in addition to our own 300 family systems. This is a real problem solved for OEMs who must provide a service solution in order to sell their value-added products!

Intel's application engineering organization is one of the largest in the world. We have application engineers specializing in everything from system software to complex peripherals. These professionals are dedicated to supporting OEMs in the pre-sales development environment. Another such group is dedicated to post-sales consulting support.

Intel's built-in quality, army of trained hardware and software application engineers, and over 75 worldwide field service locations are substantial advantages to the OEM trying to get an application off the ground and maintained in the marketplace.

VLSI Technology Leadership at the Systems Level... Today and Tomorrow

Intel is the undisputed leader in microprocessor VLSI; we invented both the microprocessor and the industry-standard MULTIBUS and are the world's largest producer of MULTIBUS board-level products.

Intel is now making a major thrust into the OEM microcomputer systems market. The fact is, we've been in the systems business for a number of years; we're the largest supplier of microprocessor development systems in the world!

Intel is uniquely positioned to offer VLSI technology leadership to OEMs at the low-risk, quick-to-market systems level. The System 380's industry-standard MULTIBUS is a guarantee of continued system compatability and expandability for years to come. With Intel systems, there are no dead ends lurking in the future for the OEM's product. An Intel system can absorb new VLSI advances and functionality almost as quickly as they become available.

Intel is committed to industry standards which lay the foundation for easy access to future technologies, future applications and future markets. For the OEM seeking to stay ahead in today's fiercely competitive markets, Intel Open Systems are the only systems to build upon.

*UNIX is a trademark of Bell Laboratories.

Specifications

Central Processor: System 380-2A, 2B, 2C models provide an 80286 General Purpose Processor and 80287 Numeric Data CoProcessor; System 380 AA, AB, AC models provide an 8086 General Purpose Processor and 8087 Numeric Data CoProcessor. Instructions are 8-, 16-, or 32-bits in length, data are 8- or 16-bits long; numeric processing (with Numeric CoProcessors) is carried out in 80-bit words.

Peripheral Interfaces: See Configuration Summary for serial port configurations; one Centronics-compatible printer interface is provided on all System 380 models.

Mass Storage: See Configuration Summary for standard configurations; all drives are 8-inch format; the Winchester controller provides complete ECC write/read checks for data integrity.

System Expansion: Eleven MULTIBUS (IEEE 796) slots for customizing and expansion using boards available from Intel and independent hardware vendors.

Environmental Specifications:
Operating: 15°C to 35°C, 20% to 80% Relative Humidity, non-condensing (26°C maximum wet bulb temperature),
Altitude: Sea Level to 8,000 feet

Regulatory Agency Specifications: Meets: UL 114—Safety; CSA 22.2—Safety; FCC Docket 20787—RFI/EMI; VDE 0871—RFI/EMI

AC Power Input: 92-126 VAC, or 184-252 VAC, 47-63 HZ (user selectable upon ordering).*†

†80286 Systems sold in three versions: A (120 VAC/60HZ), B (220 VAC/50Hz) and C (100 VAC/50Hz)
*8086 Systems sold in three versions: AA (120 VAC/60HZ), AB (220 VAC/50Hz), and AC (100 VAC/50HZ)

Configuration Summary

	System 286/380†	System 86/380*
Microprocessor	80286 (6 MHz)	8086 (5 MHz)
Numeric Coprocessor	80287 (4 MHz)	8087 (5 MHz)
RAM Memory	512KB W/ECC	384KB
...Expandable To:	10 MB using 1MB RAM boards	896 KB
Mass Storage	1 MB Diskette 35 MB Wini	1 MB Diskette 35 MB Wini
Multibus Expansion Slots	Nine @ 0.8 inch Two @ 1.2 inch	Nine @ 0.8 inch Two @ 1.2 inch
Serial Ports	Two RS-232C	One RS-232C
Expansion Power Available:		
Processor Box—		
Amps @ +5VDC	53.8	55.1
−5VDC	2.2	2.8
+12VDC	5.2	5.6
−12VDC	2.9	3.0
Total Expansion Power	377 Watts	394 Watts
Peripherals Box—		
Amps @+5VDC	26.5	26.5
−5VDC	1.3	1.3
+12VDC	2.0	2.0
−12VDC	2.6	2.6
+24VDC	3.7	3.7
−24VDC	1.6	1.6
Total Expansion Power	321 Watts	321 Watts
AC Input Power Max. Req. (Approx.)		
Processor Box—	1250 Watts	1250 Watts
Peripherals Box—	820 Watts	820 Watts

†Sold in three versions: A(120 VAC/60HZ), B(220 VAC/50Hz), C(100 VAC/50Hz)
*Sold in three versions: AA (120 VAC/60HZ), AB (220 VAC/50Hz), and AC (100 VAC/50HZ)

Documentation

Title	Literature Number
System 86/380 Overview Manual	144720-001
System 86/380 Hardware Reference Manual	172761-001
System 86/380 Installation and Maintenance Manual	144721-001
System 286/380 Installation and Operation Guide	134589
System 286/380 Hardware Maintenance Manual	134595
System 286/380 Processor Configuration Guide	135033
System 286/380 Memory Configuration Guide	135034
System 286/380 Disk Configuration Guide	134594
System 286/380 Hardware Integration Guide	134590

Manuals may be ordered directly from the Intel Literature Department.

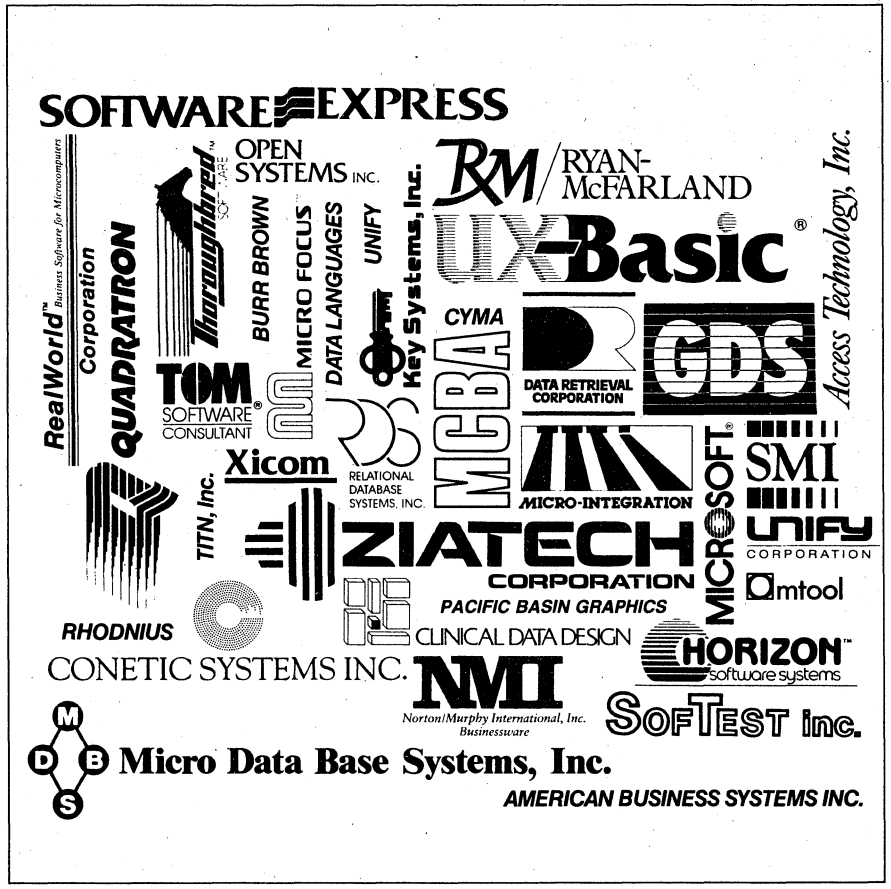
Ordering Information

System Description	Order Code
OEM Hardware System 86/380	SYS380-1A*
iRMX™ System 86/380	SYS380A-AR-Kit*
XENIX Multiuser System 86/380@	SYS380A-AX-Kit*
OEM Hardware System 286/380	SYS380-2A†
iRMX™ System 286/380	SYS380-2AR†
XENIX Multiuser System 286/380@	SYS380-2AMX†

* Sold in three versions = AA (120VAC/60Hz), AB (220 VAC/50Hz), AC (100 VAC/50Hz)
† Sold in three versions = A (120VAC/60Hz), B (220 VAC/50Hz), C (100 VAC/50 Hz)
@ Includes 4-Channel Intelligent Terminal Controller Board

THIRD PARTY SOFTWARE FOR INTEL SYSTEMS

- Over 125 Intel qualified software packages to meet your software needs
- Select from a choice of packages in most applications areas
- Support from the experts—the software manufacturers themselves
- Tested by Intel to ensure quality and reliability on Intel Systems



ORDER NUMBER: 280048-001

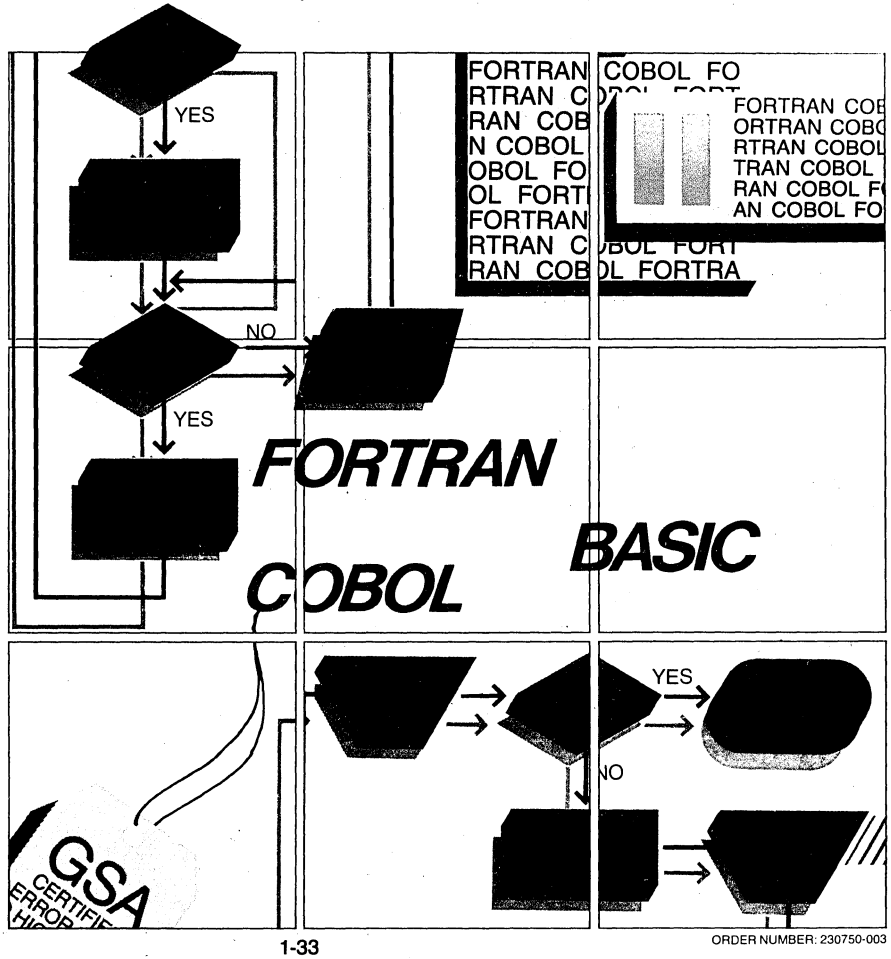
THIRD PARTY SOFTWARE FOR INTEL XENIX* SYSTEMS						
PRODUCT NAME	VENDOR	SYSTEM				AVAILABILITY
Languages		**86/310**	**286/310**	**86/380**	**286/380**	
Microsoft BASIC	Intel	X	X	X	X	Immediate
Micro Focus COBOL	Intel	X	X	X	X	Immediate
Microsoft FORTAN	Intel	X	X	X	X	Immediate
RM COBOL	Ryan McFarland		X			Immediate
SMC BASIC	SMC		X			Immediate
Softbol	Omtool		X	X		Immediate
UX BASIC	UX Software		X			Immediate
S-TRAN	SMI		X			Immediate
TOM BASIC	TOM Software		X			Q4/84
Spreadsheet						
iPlan	Intel	X	X	X	X	Immediate
20/20	Access Technology		X			Q4/84
Office Automation						
iWord	Intel	X	X	X	X	Immediate
iMenu	Intel	X	X	X	X	Immediate
Q-One	Quadratron		X			Immediate
Q-Menu	Quadratron		X			Immediate
Q-Calc	Quadratran		X			Immediate
Q-Mail	Qaudratron		X			Immediate
Q-Date	Quadratron		X			Immediate
Q-Call	Quadratron		X			Immediate
Q-Note	Quadratron		X			Immediate
Q-Form	Quadratron		X			Q4/84
DATA 3500	Tom Software		X			Q4/84
LEX	Softest		X			Immediate
Application Generator						
APPGEN	Software Express		X			Immediate
C/Tools	Conetic Systems		X			Q4/84
Graphics						
iGraph	Intel	X	X	X	X	Q4/84
PBG 200	Pacific Basin Graphics		X			Q4/84
SMC Color Graphics	SMC		X			Q4/84
Communications						
PC Link	Intel	X	X	X	X	Q4/84
3270 Bisync	Intel	X	X	X	X	Q4/84
3270 SNA	Xicom		X		X	Q4/84
X.25	T.I.T.N.		X		X	Q4/84
3780 Bisync	Micro Integration		X		X	Q4/84
HASP	Intel	X	X	X	X	Immediate
Database						
iDB	Intel	X	X	X	X	Q4/84
Informix	RDS		X			Immediate
File-it!	RDS		X			Immediate
C-ISAM	RDS		X			Immediate
Unify	Unify		X			Q4/84
MDBS III	MDBS		X			Q4/84
IDOL	SMC		X			Immediate
Progress	Data Languages		X			Q4/84

*XENIX is a trademark of Mirosoft Inc.

PRODUCT NAME	VENDOR	SYSTEM				AVAILBILITY
Accounting		**86/310**	**286/310**	**86/380**	**286/380**	
Thoroughbred Accounting	SMC		X			Immediate
Open Systems Accounting	Open Systems		X			Immediate
MCBA Accounting	MCBA		X			Immediate
APPGEN Accounting	Software Express		X			Immediate
BACs	ABS		X			Immediate
Real World Accounting	Real World		X			Q4/84
CYMA Accounting	CYMA		X			Q4/84
Complete Accounting	NMI		X			Q4/84
Manufacturing						
MCBA Manufacturing	MCBA		X			Q4/84
ProfitKey	Key Systems		X			Q1/85
A&M Manufacturing	TOM Software		X			Q4/84
Specialty Manufacturers	NMI		X			Q4/84
Medical						
MDX	Clinical Data		X			Immediate
Vertical						
Contractor Management	TOM Software		X			Q4/84
Distributor Management	TOM Software		X			Q4/84
Not-for-Profit	TOM Software		X			Q4/84
Project Management	TOM Software		X			Q4/84
Property Management	TOM Software		X			Q4/84
Public Accountant	TOM Software		X			Q4/84
Restaurant Management	TOM Software		X			Q4/84
Personnel Searcher	NMI		X			Q4/84
Magazine Circulation	NMI		X			Q4/84
Customer Profile	NMI		X			Q4/84
Trucking Dispatcher	NMI		X			Q4/84
Phototypesetting	NMI		X			Q4/84
Client Accounting	CYMA		X			Q4/84
Construction	CYMA		X			Q4/84
Chiropractic	CYMA		X			Q4/84
Orthodontic	CYMA		X			Q4/84
Dental	CYMA		X			Q4/84
Medical	CYMA		X			Q4/84
THIRD PARTY SOFTWARE FOR INTEL iRMX™ SYSTEMS						
Languages		**86/310**	**286/310**	**86/380**	**286/380**	
Microsoft BASIC	Intel	X	X	X	X	Immediate
Mark Williams C	Intel	X	X	X	X	Immediate
Graphics						
PBG 100	Pacific Basin Graphics	X	X			Immediate
Communications						
3270 SNA	Xicom	X	X	X	X	Immediate
X.25	T.I.T.N.	X	X	X	X	Immediate
3270 Bisync	Data Retrieval	X	X	X	X	Immediate
3780 Bisync	Micro Integration	X	X	X	X	Immediate
Database						
DxSystem	GDS	X	X			Q4/84
Manufacturing						
Ladder 86	Engineering Tools	X	X			Q4/84
Other						
IEEE-488	Ziatech	X	X			Immediate
Driver to Data I/O subsystem	Burr Brown	X	X			Q4/84

XENIX* LANGUAGES

- COBOL, BASIC and FORTRAN support for Xenix-based systems
- Conformation to international standards: ANSI 77 subset FORTRAN, ANSI X3.23 1974 COBOL to Federal High Level and ANSI X3.60—1978 subset BASIC
- Powerful microcomputer extensions to ANSI standards
- Easy porting of mainframe and minicomputer applications to micro environment
- Intel 80287 math coprocessor support
- Worldwide service and support organization



*XENIX is a trademark of Microsoft Corporation.

ORDER NUMBER: 230750-003

High-level Language Support for XENIX-Based Systems

Intel's XENIX operating system, available for component, board, or system-level integration, is a multi-user operating system well suited for both technical and commercial interactive applications. Typical applications include small business systems, software development/engineering workstations, distributed data processing and graphics.

For OEM and end-user application development on XENIX, Intel has provided three industry-standard, high-level languages—FORTRAN, COBOL and BASIC—with which to build microcomputer-based solutions for systems products or component and board-level applications. XENIX BASIC, FORTRAN and COBOL accommodate easy porting of existing mainframe and mini-based applications to the micro environment.

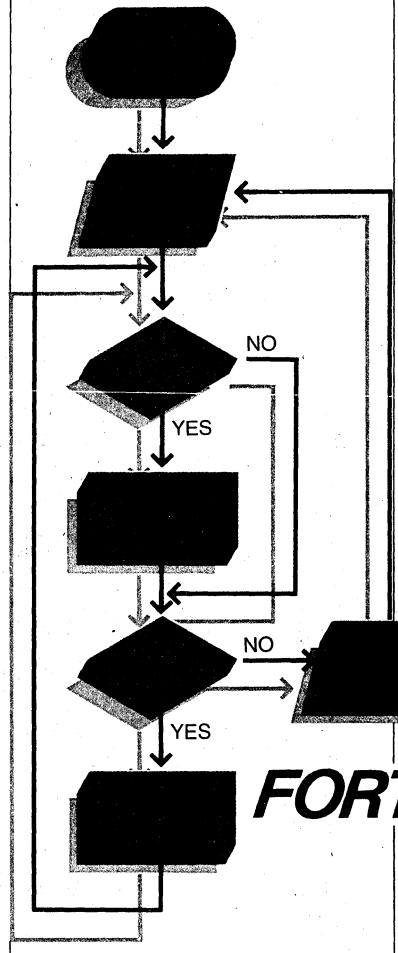
XENIX FORTRAN for Scientific and Technical Applications

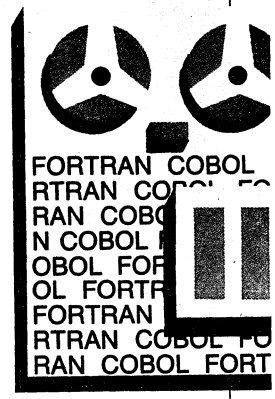
FORTRAN is the most popular programming language for scientific and numerical applications. There are thousands of existing FORTRAN programs and subroutines written in mainframe and minicomputer environments, most of which can be ported to a micro environment via Intel's offering of Microsoft FORTRAN.

Compliance with the X3.9 1978 ANSI standard for FORTRAN at the subset level ensures portability with minimal source code modifications. By moving to a microcomputer-based system, you lose none of your mainframe and mini-developed software investment.

Speed and Accuracy Where They're Needed

Scientific, math-oriented applications usually require fast, highly accurate processing. XENIX FORTRAN delivers accuracy with double-precision arithmetic which handles numbers containing 15 significant digits.





High speed results from XENIX FORTRAN support of the Intel 80287 floating point coprocessor, as well as from an extensive subroutine library, which includes subroutines for 16- and 32-bit integer arithmetic and 32- and 64-bit floating-point arithmetic. Because of XENIX FORTRAN's 80287 math coprocessor support, some programs written in XENIX FORTRAN will execute from two to four times faster than their minicomputer counterparts.

Calls to "C" and MS MACRO Assembler are possible, making it easy to interface non-standard peripherals to XENIX FORTRAN programs.

FORTRAN

XENIX COBOL for the Micro Environment

Intel's offering of Microfocus COBOL is a mainframe-caliber compiler for ANSI 1974 COBOL programs, enabling XENIX-based systems to compile and run existing COBOL programs with minimal source code modification. XENIX COBOL also contains features specifically aimed at facilitating the interactive

*XENIX is a trademark of Microsoft Corporation.

BASIC

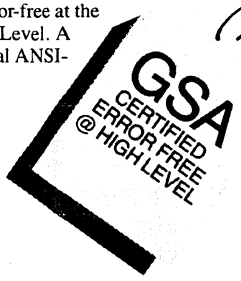
FORTRAN COBOL F
ORTRAN COBOL FO
RTRAN COBOL FOR
TRAN COBOL FORT
RAN COBOL FORTR
AN COBOL FORTRA

program development of new applications in a microcomputer environment.

These features include a facility for dynamically loading sub-programs from disk as required which effectively removes limits on the size of the application code that can be run. XENIX COBOL augments the functionality of the ANSI standard with additional compiler features, such as interactive screen-handling, that further increase convenience and programmer productivity.

Users can license a separate run-time support package. This enables OEMs to pass COBOL applications onto customers at a much lower cost than that involved in transferring full COBOL packages.

XENIX COBOL is one of only eleven COBOL compilers in existence—and the only one for microcomputers—that has been GSA-certified as error-free at the High Level. A special ANSI-defined communications module provides the user with a standard mechanism for program-to-program message-passing in multi-user networks such as those found in an "office of the future" settings.



Forms-2™ Support for Screen-Painting

XENIX COBOL supports FORMS-2, a powerful visual programming tool that speeds the creation of programs involving interactive screen-handling. In an extremely user-friendly environment, the user "paints" a form on the screen, and FORMS-2 generates the COBOL source code to support it. FORMS-2 results in greatly improved programmer productivity in a microcomputer, screen-building environment.

XENIX BASIC for Maximum Flexibility

Intel's offering of Microsoft BASIC opens a whole window of applications to the XENIX user. Since their BASIC is the same as that used on MS-DOS* based machines, most programs written for MS-DOS can now run on XENIX unchanged. When developing your own programs, BASIC is simple and easy for quick prototyping, yet complete enough for total development. Conforming to the ANSI X3.60 1978 subset standard, BASIC also has powerful extensions, 16 significant digit Double Precision floating point arithmetic, 80287 support, and assembly languages routine calling capabilities. From using applications to designing your own programs BASIC is easy, complete, and extremely flexible.

COBOL

Worldwide Service and Support

All XENIX systems are fully supported by Intel's worldwide staff of trained hardware and software engineers. Complete documentation is provided for all operating systems and application software languages, as well as for system hardware components. The XENIX and XENIX Languages warranty includes Hotline support, Software Updates, and Subscription Service.

Total Solutions for Interactive, Multi-User Applications

Intel's XENIX-based systems offer the most complete solutions for interactive, multi-user applications requiring fast, accurate throughput and a friendly programming environment. XENIX is complemented by industry-standard, high-level languages with which OEMs can create flexible and open end-user systems.

XENIX languages are completely portable—from one level of integration to another (chip to board to system).

Intel is paving the way into the future of VLSI and pioneering VLSI-based systems. We are committed to providing customers with smooth, uninterrupted application development on the latest VLSI-based systems - today and tomorrow.

Specifications

Required Hardware:	Required Software:	
• Any iAPX 286 based or iSBC® 286 based system including Intel's 286/300 family and iDIS systems • 196 KB memory • Two floppy disks or one hard disk • One 8" double-density or 5.25" double-density floppy disk drive for distribution of media	• Intel's XENIX 286 Operating System • Purchase of any XENIX Language requires signing of Intel's OEM License Agreement (OLA)	

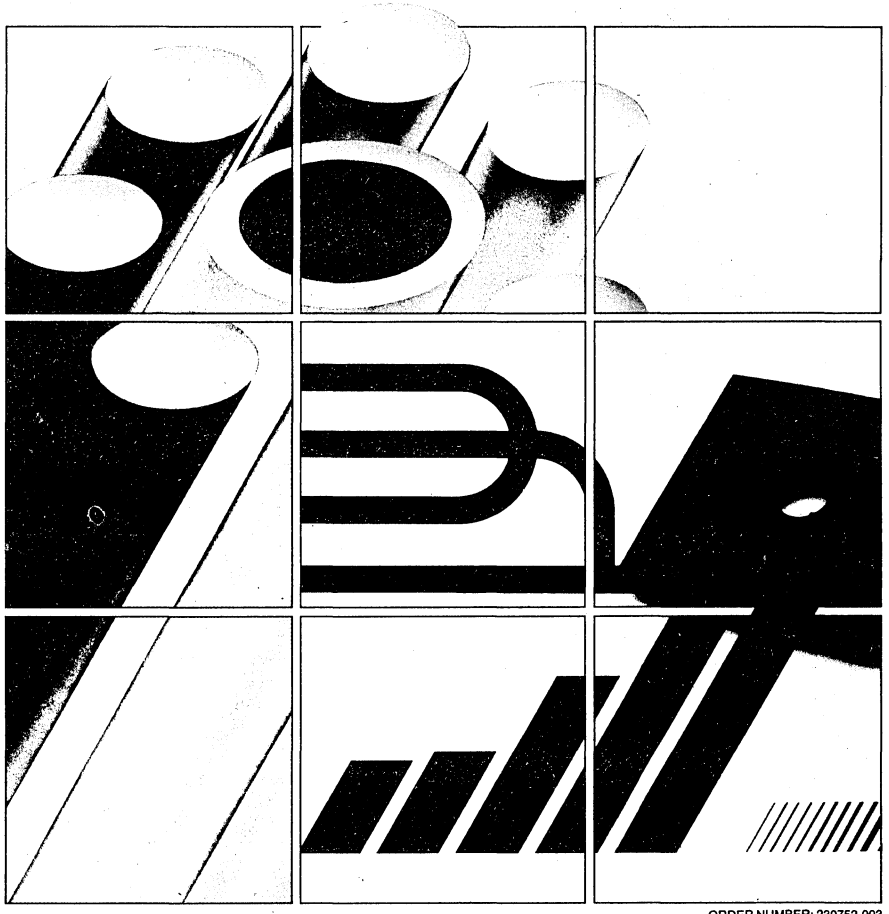
Ordering Information

Language	Order Code	Product Contents	Warranty
COBOL	XNX 2867	One 8" diskette and one 5.25" diskette Level II COBOL Language Reference Manual—122158 Level II COBOL Operating Guide—122159 Forms II Utility Manual—122160 Level II COBOL Pocket Guide—122161	**90 days:** Software Updates, Subscription Service
	XNX 2868	Incorporation Fee for passing through the COBOL Runtime System	
FORTRAN	XNX 2862	One 8" diskette and two 5.25" diskettes Fortran Reference Manual Fortran User's Guide	**90 days:** Software Updates, Subscription Service
BASIC	XNX 2865	One 8" diskette and one 5.25 diskette BASIC Reference Manual BASIC User's Guide	**90 days:** Software Updates, Subscription Service

FORMS-2 is a trademark of Micro Focus.

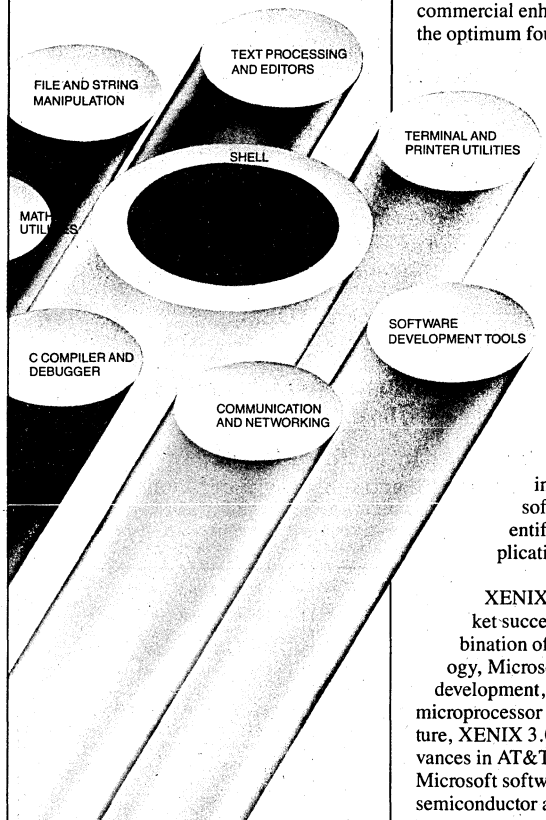
XENIX* 3.0 OPERATING SYSTEM

- XENIX 3.0 Industry Standard Multiuser Operating System
- Fully licensed version of the UNIX† operating system optimized for the Intel 80286 processor
- Leading edge microprocessor implementation of UNIX, fastest floating point performance on a microprocessor
- Important commercial OEM enhancements
- Supports multiple levels of integration: components, boards and systems
- Supported by Intel's worldwide post-sales service and support organizations



*XENIX is a trademark of Microsoft Corporation
†UNIX is a trademark of Bell Laboratories.

ORDER NUMBER: 230752-003



XENIX 3.0—Industry Standard Multiuser Operating System

Intel's XENIX 3.0 Operating System for the 80286 is a fully-licensed derivation of Bell Laboratories' UNIX System III. XENIX 3.0 includes not only all the functionality of UNIX System III, but also powerful enhancements from Microsoft and Intel that meet the needs of the commercial OEM.

The Best Foundation for Building OEM Solutions

XENIX 3.0 provides the OEM with a complete software base on which to build value-added functionality. It includes the operating system, the C language, text processors, development tools, system accounting and security features, and commercial enhancements that make it the optimum foundation for OEM application software solutions.

*XENIX is a trademark of Microsoft Corporation.

XENIX: Portable, Flexible, Powerful

XENIX has become the industry-standard microcomputer operating system for interactive, multi-user applications. It has gained wide popularity in applications such as distributed data processing, business data processing, word processing, software development, scientific and engineering applications, and graphics.

XENIX has achieved this market success through a solid combination of UNIX system technology, Microsoft value-added product development, and Intel's experience in microprocessor technology. In the future, XENIX 3.0 will benefit from advances in AT&T UNIX technology, Microsoft software technology and Intel semiconductor and system technology.

XENIX is also an extremely powerful operating system, providing the applications programmer with a wealth of development tools and utilities for bringing OEM products to market quickly.

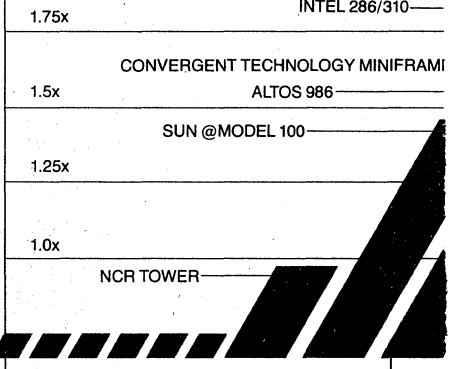
XENIX 3.0: Leading Edge UNIX Performance on a Micro

As the first UNIX operating system derivative optimized for the iAPX 286, XENIX 3.0 alone can take full advantage of the 80286's unique features:

On-chip memory management and protection provides two key advantages for XENIX 3.0 over other microprocessor UNIX implementations. First, on-chip memory management and protection drastically reduces the overhead in accessing system memory as compared to the usual separate memory management unit. With this functionality right on the chip, the operating system works more smoothly and efficiently.

Second, on-chip memory management and protection circuitry ensures that each version of XENIX 3.0 will be very compatible with every other version. This heretofore impossible level of compatibility aids OEM, software developers, and end users due to the wider availability of compatible software.

Advanced microprocessor architecture provides pipeline processing, wherein a continual flow of instructions is kept in the CPU queue, results in throughput several times faster than the fastest competing microprocessor.



Fast floating point processing is due to XENIX 3.0 support of the Intel iAPX 287 math coprocessor. Floating point processing delivers throughput that is an order of magnitude faster than non-floating point processing. Extra high processing speeds are needed in applications such as data base processing, commercial data reduction and graphics.

Faster, More Reliable Still When Teamed with Other Intel Systems Components

The throughput enhancements in the XENIX 3.0 software are pushed to even greater speeds by special hardware architecture in Intel's systems and board products.

MULTIBUS® System Architecture is the industry-standard system bus. It accommodates any of the special-purpose Intel iSBC® boards, as well as a multitude of third party Multibus boards and standard peripherals, for easy system expansion.

iLBX™ (Local Bus Exchange) is an Intel hardware innovation that increases the amount of local memory accessible by the operating system to significantly improve system throughput.

Error Correction Circuitry (ECC) automatically detects and corrects soft errors in RAM. This on-board, self-correction facility reduces errors and further underscores data integrity.

See Intel benchmark series

A Faster Operating System Means Market Leadership

The combination of the industry's most widely accepted operating system for multi-user, interactive applications with the industry's fastest and most advanced microprocessor gives the OEM a far superior price/performance ratio than is available through other options. The result for the OEM: market leadership due to the ability to more attractively price products based on superior performance.

XENIX 286 combines UNIX technology from these organizations.

XENIX 3.0: The Best of Everything

The XENIX 3.0 Operating System contains the best of many vendors' UNIX/XENIX development efforts during the last ten years (see Fig. above). We have taken the best features of many UNIX versions—ease of use, flexibility, performance, security, reliability—and added our own enhancements (not the least of which is compatibility with the iAPX 286) to make XENIX 3.0 the optimum software foundation for the commercial OEM.

Superior Date Reliability and Integrity

XENIX 3.0 contains enhancements to provide extremely high data reliability and integrity, particularly important to the OEM who is adding value to a system product. The following enhancements in XENIX 3.0 contribute to uniformly reliable data at all stages of application development.

Automatic disk recovery is an improvement of the UNIX file system that allows automatic recovery of the file system in the event of unexpected system shutdown.

Record and file locks arbitrate multiple-access requests to the same record or file, allowing the programmer to extend locks to a single record, group of records or the entire file. This is important in multi-user applications to prevent two or more users accessing and updating the same information simultaneously.

XENIX System Analysis Test (XSAT) is a complete hardware-software diagnostic package included with all Intel integrated system products. XSAT provides a total analysis of a XENIX-based system, ensuring reliability even after the OEM configures new drivers into the system.

Tools for Easy System Configuration

In addition to increased data reliability measures, XENIX 3.0 has been functionally enhanced for easier system configuration. An interactive configuration utility allows the user to specify device drivers, disk buffers, memory size, etc., making it easy for the OEM to meet unique design requirements. XENIX 3.0 includes over 12 device drivers for high-speed controllers.

Friendlier Interface

The standard UNIX human interface has been enhanced in XENIX 3.0, with the

addition of vi, a full-screen editor, for easier and faster application development.

The XENIX C shell augments the capabilities of the standard UNIX shell with the ability to maintain histories of invoked processes and provide the alias feature, saving re-keying of often-used commands. XENIX 3.0 also provides the visual shell, a menu driven command interpreter which makes full use of the screen to display status and environmental information to the user. It has a built-in HELP facility and allows users to add new applications to the menu.

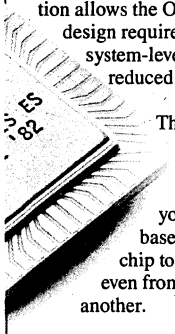
Intel's Open Systems Approach

Intel believes that system components—hardware or software—should be fully compatible with other family members at any level of integration and open to future VLSI advancements. XENIX 3.0 was designed to be part of the Open Systems concept.

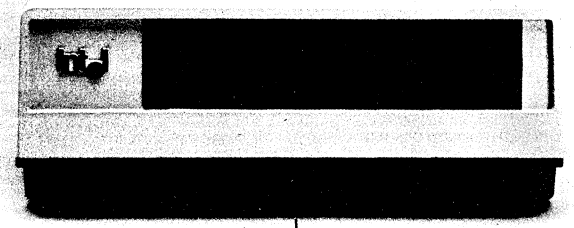
Portability from Chip to Board to System

Intel's XENIX 3.0 Operating System is available for and fully compatible across Intel component, board and system designs, something that no other XENIX version offers.

Such portability gives OEMs the flexibility to choose the most appropriate and profitable level of integration for their applications. Component-level integration allows the OEM to meet unique design requirements; board and system-level integration afford reduced time to market.



There is no loss in software development investment as your needs change, since you can port XENIX-based applications from the chip to the system level or even from one Intel processor to another.



Open to Still Greater Configurability through Third-Party Software and Hardware

XENIX 3.0 users can tap into an extensive base of existing third-party languages and application packages for almost endless versatility in system configurability. There are hundreds of such packages available today with many more on the way. To assure the availability and quality of these packages on our systems, we have the Independent Software Vendor Program. Through this activity, software vendors are given Intel systems as well as technical assistance to aid them in porting their packages. The resulting product is thoroughly evaluated by Intel prior to certification for operation on our system products.

Superior Documentation

In line with the OEM orientation of the Intel hardware and software combination, the documentation for Intel's XENIX 286 product provides excellent support for system builders. In addition to the mature UNIX documentation from AT&T and the value-added feature documentation by Microsoft, Intel adds a wealth of publications aimed at helping the OEM to successfully launch XENIX 3.0 based products.

Worldwide Support and Service

XENIX 3.0 customers can take advantage of Intel's worldwide staff of trained hardware and software engineers in contracting for application design assistance. A liberal warranty, including software updates and a technical newsletter, follows the sale. Once the warranty expires customers can choose from a variety of support contracts.

Intel offers complete training on the XENIX 3.0 Operating System as well as the iAPX 286 processor and associated hardware.

Intel, The Technological Leader...

Intel is committed to pushing the frontiers of VLSI design to their ultimate limits. In the process, we move our customers along the technology curve without interruptions in application development or expensive mid-stream architecture changes.

Intel started the micro revolution with the 4004 and has been the market leader with every generation of advanced processors since.

Systems and system software are a natural for us: who better knows the pieces and how to make them work together?

...In Total Solutions

The XENIX 3.0 Operating System fully exploits the iAPX 286, the fastest and most sophisticated microprocessor on the market. No other processor/operating system combination will give OEMs a faster and more economical path to getting systems and applications on the market.

Intel has always been first with the latest and most advanced VLSI and now with system software tailor-made for Intel VLSI. Because we're there first, our customers are first in their respective markets with state-of-the-art OEM and end-user products.

XENIX 3.0 includes support for the following Intel Systems, single board computers and processors.

- System 286/310
- System 286/380

- iSBC® 286/10 Processor Board
  —16mb of addressing
  —On-chip memory protection
- CX Series RAM board
  —ECC (Error Correction Circuitry)
  —iLBX™ (Local Bus Extension)
- iSBC 215 Winchester Controller
- iSBX 218 Floppy Controller
- iSBC 534 Serial I/O Expansion Board
- iSBC 544 Intelligent Serial I/O Expansion Board
- iSBC 188/48 8-channel Serial I/O Expansion Board
- iSBC 552 Ethernet Controller Board
- iSBX 217 Tape Controller Board

- 80286 Central Processor
- 80287 Fast Floating Point Processor

Documentation

Documentation Includes:

- Overview of the XENIX 286 Operating System
- XENIX 286 Installation and Configuration Guide
- XENIX 286 System Administrator's Guide
- XENIX 286 Communications Guide
- XENIX 286 Visual Shell User's Guide
- XENIX 286 User's Guide
- XENIX 286 Reference Manual
- XENIX 286 C Library Guide
- XENIX 286 Programmer's Guide
- XENIX 286 Device Driver Guide
- XENIX 286 Text Formatting Guide

Text Books

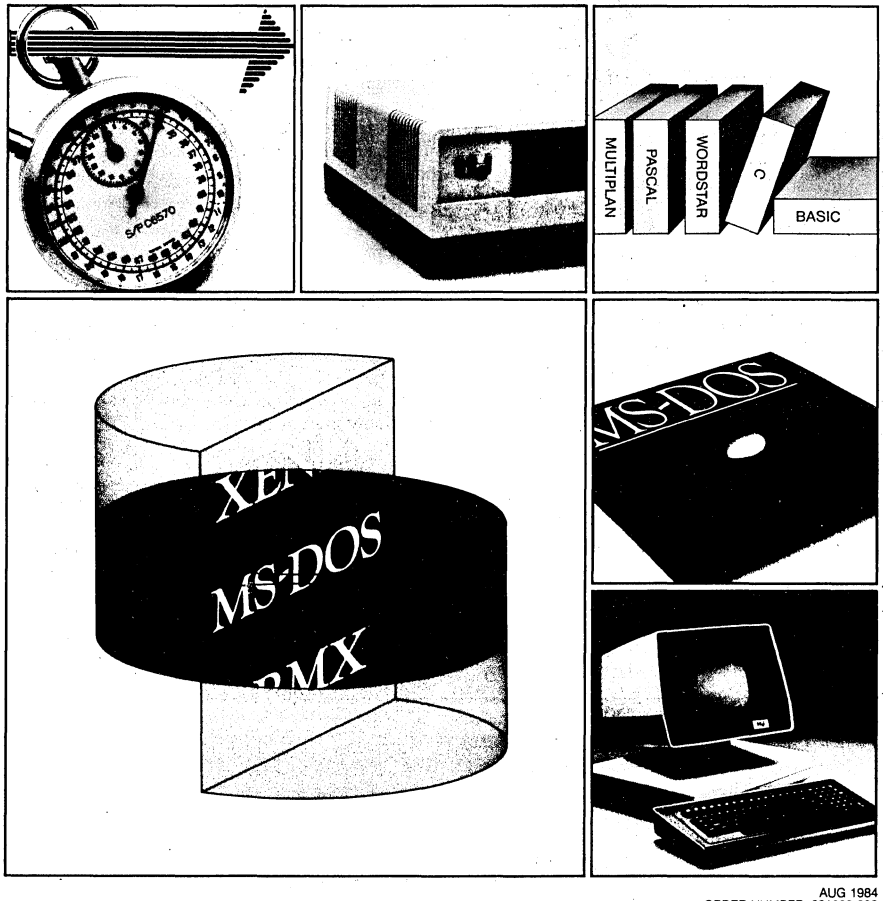
The UNIX Book—Banahan & Rutter
The UNIX System—Bourne
The UNIX Operating System—Kaare
Understanding UNIX: A Conceptual Guide—Groff & Weinberg
The UNIX Programming Environment—Kernighan & Pike
Introducing the UNIX System—McGilton & Morgan
A Practical Guide to the UNIX System—Sobell
A User Guide to the UNIX System—Yates & Thomas
A Business Guide to the UNIX System—Yates and Emerson

Ordering Information

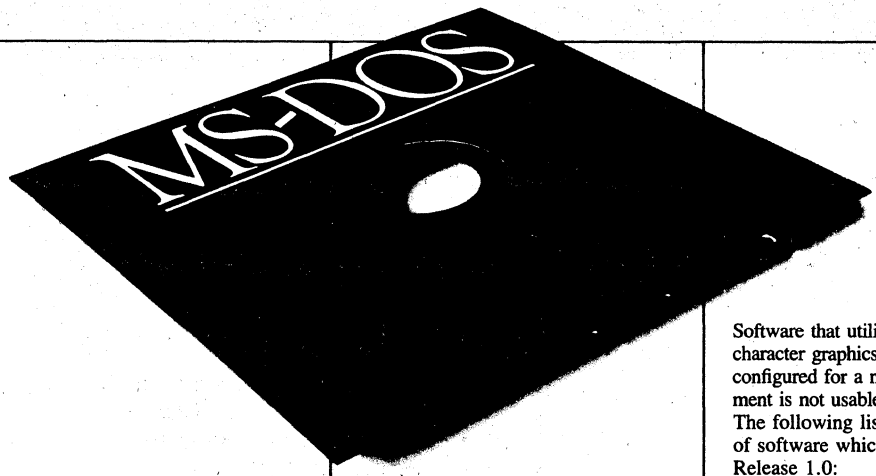
XNX 286 HRO	XENIX Object Software (8" double side, double density) plus license rights
XNX 286 KRO	XENIX Object Software (5¼" double-sided, double density) plus license rights
XNX 286 RF	Software Incorporation Fee
SYS 310-17X	System Kit including System 310-17 and XENIX Software
SYS 310-17MX	System Kit including System 310-17, XENIX Software, 6 user support
SYX 286 RO	License rights extension for system customers
SYX 286 RF	System incorporation fee

310 MS-DOS(2.11) OPERATING SYSTEM RELEASE 1.0

- **A fully licensed port of MS-DOS* 2.11 for Intel's 310 family of microprocessor-based systems**
- **Creates a high-performance, high-capacity MS-DOS system**
- **Supports 86/310 models and 286/310 models**
- **Three supported environments:**
  Alternate iRMX™/MS-DOS
  Alternate XENIX*/MS-DOS
  Exclusive MS-DOS
- **Partitioned Winchester option**
- **Provides immediate access to most MS-DOS applications and languages**



AUG 1984
ORDER NUMBER: 231033-002



MS-DOS for the 310

The 310 MS-DOS product is a complete and fully licensed port of Microsoft's MS-DOS 2.11 operating system for Intel's 310 family of 8086 and 80286 microprocessor-based systems. Most often purchased as a complementary operating system for 310 systems running XENIX or iRMX, 310 MS-DOS delivers immediate user access to most off-the-shelf MS-DOS applications and languages, while creating a high-performance, high-capacity MS-DOS system. 310 MS-DOS is the 310's lowest priced operating system and is fully supported by Intel.

310 MS-DOS is a single-user/single-terminal operating system. When running MS-DOS, exactly one 310 terminal is usable.

Your Window to the World of PC Software

The addition of MS-DOS to the 310's operating system options adds a new dimension of versatility to the system. 310 MS-DOS allows a user to run, without modifications, many off-the-shelf applications and languages written for the burgeoning world of personal computers. The following list is representative of the software usable in the 310 MS-DOS environment:

dBASE-II*	(Ashton-Tate)
WORDSTAR	(Micropro)
CALCSTAR	(Micropro)
Multiplan	(Microsoft)
C Compiler	(Microsoft)
BASIC Compiler	(Microsoft)

In general, any software is usable with 310 MS-DOS provided it meets the following requirements:

- Labeled "MS-DOS"
- Does not violate standard MS-DOS operating system interfaces
- Configurable for a standard ASCII terminal

* Requires dBASE-II version 2.41a (or higher) for 286/310 operation.

Software that utilizes bit-mapped or character graphics and cannot be configured for a non-graphics environment is not usable with 310 MS-DOS. The following list is representative of software which cannot be used in Release 1.0:

Lotus 1-2-3	(Lotus)
VisiOn	(VisiCorp)
PFS:Graph	(Software Publishing Corp.)
PFS:File	(Software Publishing Corp.)

High Performance MS-DOS

The 310 MS-DOS operating system transforms Intel's 86/310 and 286/310 systems into high-performance, high-capacity MS-DOS engines. When configured with similar memory, a 310 MS-DOS system will out-perform an IBM PC-XT dramatically.

Configurable to a maximum of one megabyte of main memory, two floppies and two Winchester disks**, 310 MS-DOS delivers capacities previously unavailable to the MS-DOS user.

An Alternate Operating System

310 MS-DOS is a single-user/single-terminal operating system and may co-exist with the 310's real-time multitasking operating system iRMX, or the commercial multiuser operating system XENIX. Three configurations are supported in Release 1.0:

**See System 310 configuration specifications for floppy and Winchester disk options and capacities.

1. Alternate: 310 MS-DOS and the 310's other operating system (either iRMX or XENIX) are booted and shut-down as desired. In this environment, 310 MS-DOS does not use the 310's Winchester in any way, and must be booted from the floppy.
2. Alternate/Shared: As above except the 310's Winchester has been divided into two partitions and 310 MS-DOS "owns" one of the partitions. In this environment, 310 MS-DOS is normally booted from the Winchester and may access both the floppy and the Winchester's MS-DOS partition. Partitioning instructions are provided with 310 MS-DOS.
3. Exclusive MS-DOS: 310 MS-DOS is the only operating system used on the 310. In this environment, 310 MS-DOS possesses the 310's Winchester disk(s) entirely.

Hardware Configurations

The 310 MS-DOS product can be used with all configurations of the 310 having the iSBC® 215g/218a controllers (essentially all configurations having a Winchester disk). The operating system is available for both the 8086-based and the 80286-based configurations of the 310. 310 MS-DOS runs in 8086-compatibility mode on the 286/310 systems.

The 310 MS-DOS operating system supports all standard RS-232 ASCII terminals. In addition, MS-DOS permits user-written, user-installable device drivers, allowing the user to tailor the system to support the enhanced features of any RS-232 terminal. The procedures for writing and installing device drivers are documented in the MS-DOS Programmer's Reference Manual.

An Open System for Easy Integration

System 310's Multibus® architecture allows the OEM to select from among 1000 Multibus-compatible devices. MS-DOS allows an OEM to write and install device drivers.

Together, these features permit an OEM to easily integrate hardware devices and their software drivers into a 310 MS-DOS system.

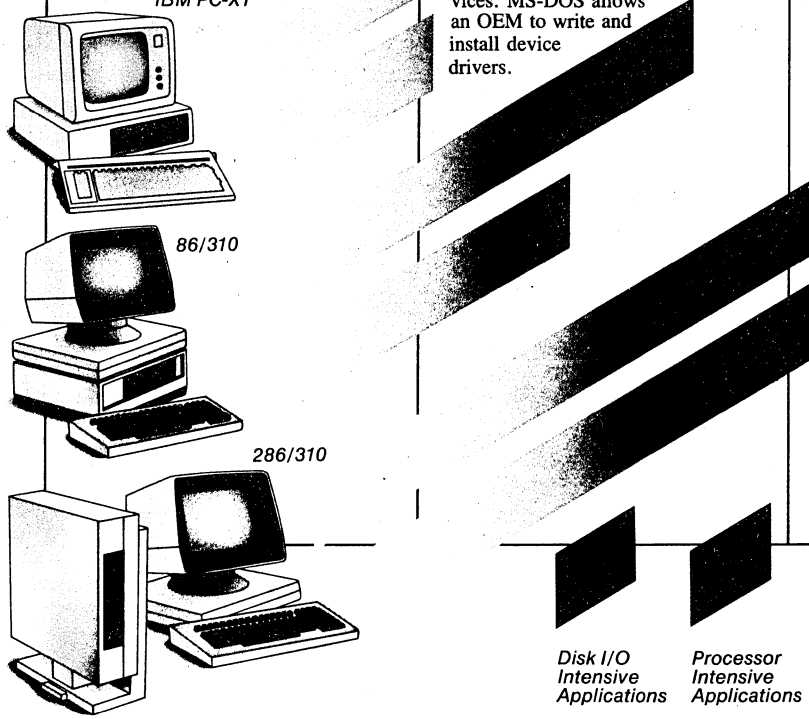
Packaging

The 310 MS-DOS product is available as two sets of diskettes, one for 86/310 systems, and one for 286/310 systems. Each set contains four diskettes.

- 310 MS-DOS System Diskette
- MS-DOS Utilities Diskette
- XENIX Bootload Installation Diskette (for XENIX/MS-DOS partitioned Winchester configurations)
- iRMX Bootload Installation Diskette (for iRMX/MS-DOS partitioned Winchester configurations)

Each package also contains the following documentation:

- 310 MS-DOS Installation Guide
- MS-DOS User's Guide
- MS-DOS Programmer's Reference Manual



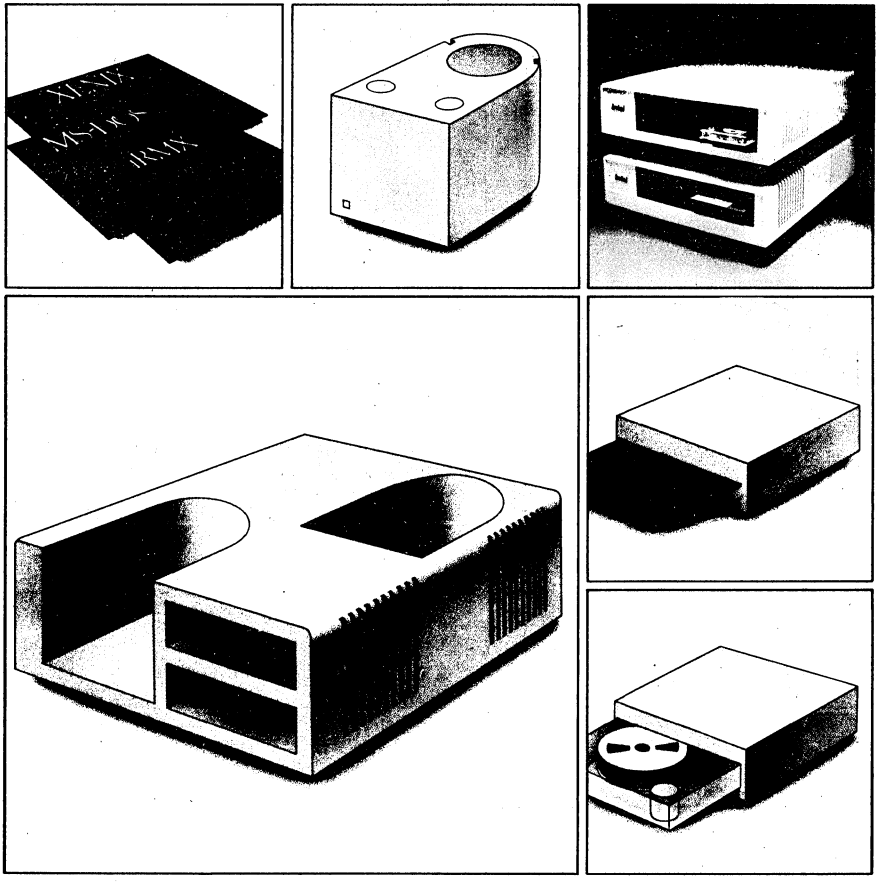
Ordering Information

310 MS-DOS can be ordered under the following product codes:

DOS86SU	MS-DOS End User Object Diskettes for the 86/310 system. Consists of the four diskettes and documentation described above. Class I software: for use (execution) by the purchaser only.
DOS286SU	MS-DOS End User Object Diskettes for the 286/310 system. Consists of the four diskettes and documentation described above. Class I software: for use (execution) by the purchaser only.
DOS86RI	MS-DOS OEM Pass-through Object Diskettes for the 86/310 system. Consists of the four diskettes and documentation described above. Class III software: For resale by the OEM purchaser.
DOS286RI	MS-DOS OEM Pass-through Object Diskettes for the 286/310 system. Consists of the four diskettes and documentation described above. Class III software: For resale by the OEM purchaser.
DOS86RO	MS-DOS Class III Incorporation License. Permits the licensee to modify, reproduce, and resell 310 MS-DOS. This is a one-time-only fee. Includes the four diskettes and documentation described above.
DOS286RO	MS-DOS Class III Incorporation License. Permits the licensee to modify, reproduce, and resell 310 MS-DOS. This is a one-time-only fee. Includes the four diskettes and documentation described above.
DOS86RF	MS-DOS Incorporation Fee. Royalty payments are made via this product code for each copy of 310 MS-DOS reproduced under the above DOS86RO Incorporation License.
DOS286RF	MS-DOS Incorporation Fee. Royalty payments are made via this product code for each copy of 310 MS-DOS reproduced under the above DOS286RO Incorporation License.
DOSMANUG	MS-DOS Documentation. Consists of the three manuals described above. Typically purchased with the RO and RF products.

311 PERIPHERAL EXPANSION SUB-SYSTEM

- Designed to complement Intel's System 310 and System 380 families
- 3 full-height peripheral bays, offering
  - □ tape back-up and archiving
  - □ high capacity Winchester drive expansion
  - □ 320KB floppy drive
- High degree of configurability and upgradability
- Worldwide service and support from Intel



September 1984
Order Number: 270023-001

311 Peripheral Expansion Sub-System

Intel's 311 is a flexible peripheral expansion sub-system for the Intel 310 and 380 product families. With three full-height 5.25" bays, the 311 provides 310 and 380 users with a wide range of peripheral expansion options that include tape, Winchester, and floppy drives.

45MB Streaming Tape Back Up

The 5.25" half-height streaming tape option provides the user with 45MB (unformated) of back-up capacity. Thus, a 40MB Winchester disk drive can be quickly, conveniently, and reliably backed up with a single tape cartridge. The tape option also presents the user with an extremely convenient removable archiving capability. One 45MB tape cartridge alleviates the need to use hundreds of floppy diskettes.

Increased Disk Storage Capacity

Users requiring higher Winchester disk storage capacities than those provided by a host system now have the option of adding two full height 5.25" Winchester disk drives. Configured with two 40MB Winchesters, the 311 provides an additional 80MB of storage.

5.25" Floppy Drive

The 311 provides a half height 5.25" 320KB floppy drive option for the System 380 user, or for the System 310 user who needs a second floppy drive.

High Degree of Configurability and Upgradability

The 311 can be configured to any combination of peripherals desired by OEMs. The 311 can be ordered in its maximum configuration, with two full height Winchester drives, a half height tape and a half height floppy, or in any subset of that configuration (for example, a "tape only" configuration or a "Winchester disk/tape" configuration). OEMs can also order the chassis-only configuration and integrate their own peripheral devices.

The 311 allows System 380 users to finally take advantage of 5.25" peripheral technology. The "Wini/floppy/tape" and the "Wini/Wini/floppy/tape" configurations provide a full complement of peripherals that a System 380 user requires.

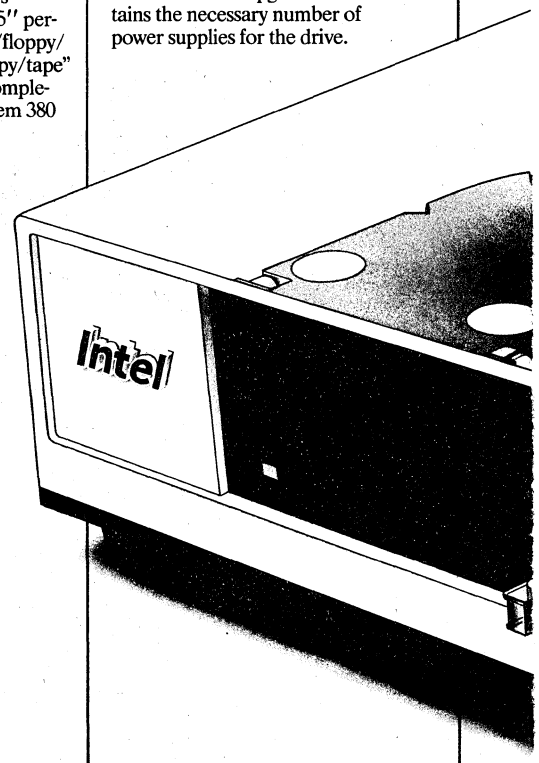
Users who need to upgrade their existing System 311 with an additional peripheral device will be able to do so conveniently with the 311 upgrade kits. There is a tape kit, a Winchester disk kit, and a power supply kit (for OEMs who want to integrate their own peripheral devices). These kits are complete with all the power supplies, cables, and documentation necessary for integration.

A Complete Solution

The 311 provides the OEM with a complete solution. It comes with all the necessary external and internal cables, power supplies and documentation that are necessary to connect the 311 to its host system. If the 18" external cables are not sufficient, a set of 5.5' cables can be ordered separately. All the configurations that have a tape drive come with an iSXM217C tape controller kit (if necessary) and a tape cartridge.

The Power Supply

The 311 power system consists of five independent power supplies. The fully configured "Wini/Wini/floppy/tape" 311 uses all five power supplies, whereas the "tape only" and the "Wini only" configurations require only two. Each drive upgrade kit contains the necessary number of power supplies for the drive.



Additionally, a power supply kit is also available. Each power supply provides 2.5A of +12V, 2.5A of +5V and 0.1A of −12V. To reconfigure the power supply between 115V and 220V requires one jumper change.

Worldwide Compatibility

The 311 adheres to a broad set of worldwide safety and signal radiation regulations. It meets FCC standards for radio frequency interference (RFI) and electromagnetic interference (EMI), as well as complying with the UL, CSA, and the European TUV and IEC-435 safety regulations.

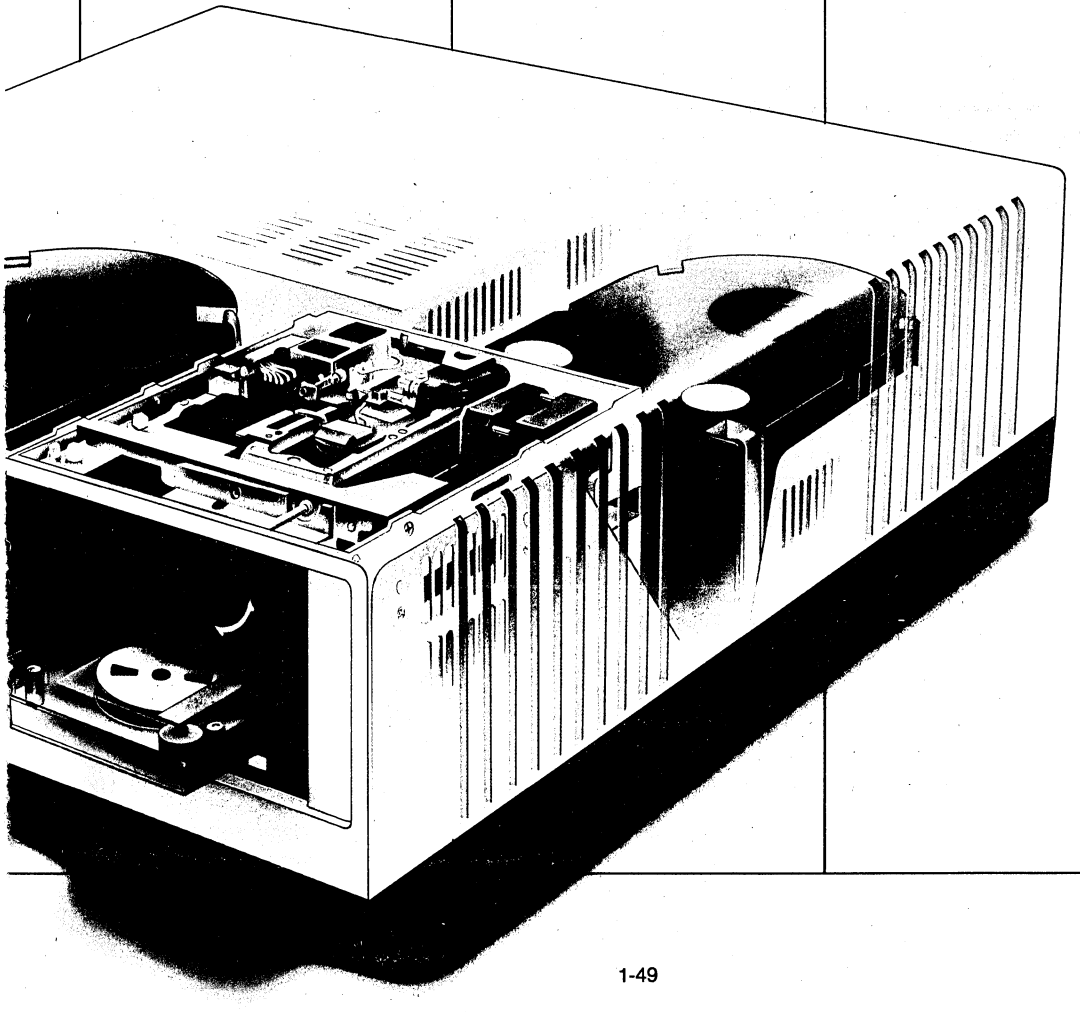
Environmental Specification

The 311 adheres to the same set of environmental standards as its host 310 and 380 systems.

Worldwide Service

All hardware is warranteed for 90 days. Maintenance contracts are available, or customers can opt for service available on a per-call basis. If on-site repair is not required, 48-hour factory repair is available as well as our economical direct-return service. Additionally, "Open Systems" service covers non-Intel portions of the system as well as Intel supplied parts.

Intel's service organization has over 80 offices worldwide.



Documentation

The 311 Peripheral Expansion Sub-System Service Manual	#134923-001
iMDDX Test Reference Manual: MD217C	#135196-001
iSXM217C Hardware Installation Guide	#135253-001

Ordering Information

The 311 product code depends on a set of three variables: the peripheral drive combination, the power supply (115V or 220V), and host controller (215G/217C/218A or 214). The following table can help determine the product code:

Peripheral Combination				Power Supply		Host Controller	
W	W	F	T	115V	220V	215G/217C/218A	214
				A	B	01	01
			x	A	B	02	12
x				A	B	03	13
x			x	A	B	04	14
x		x	x	A	B	05	15
x	x			A	B	06	16
x	x		x	A	B	07	17
x	x	x	x	A	B	08	18

Product Code □ □□

□ A or B for power type

□□ two-digit peripheral code

The product code is SYS311-PC, where P is a choice in the Power Supply column and C is a choice in the Host Controller column. For example, the product code for a "tape only" 311 with 115V power supply interfacing with a 217C tape controller is SYS311-A02. The product code for a "Wini/floppy/tape" 311 with a 220V power supply interfacing with an iSBC 214 controller is SYS311-B15.

intel®

RELIABILITY REPORT

RR-52

July 1984

SYSTEM 310 RELIABILITY

CLEONE HAWKINSON
OREGON SYSTEMS QUALITY AND RELIABILITY

Order Number 280044-001

SYSTEM 310 RELIABILITY REPORT

CONTENTS

SYSTEM 310 RELIABILITY REPORT

RELIABILITY: THE ABILITY OF A PRODUCT TO CONSISTENTLY OPERATE FAILURE FREE IN THE ENVIRONMENT FOR WHICH IT WAS DESIGNED

SUMMARY

This report describes the Intel System 310 microcomputer and presents the results of the system's reliability evaluation. This report also reviews the component, board, and system Quality Assurance and Reliability qualification programs.

Five configurations of the System 310 were evaluated: the SYP310-1, -2, and -3 (8086-based product) and the SYP310-4 and -17 (80286-based product). These System 310 products have passed a series of reliability tests that include accelerated life, environmental, (temperature, humidity, shock, vibration, altitude) and shipping tests. A summary of the Mean Time Between Failures (MTBF) data is below. Supporting data are presented in the text.

Configuration	Goal MTBF 35°C (in hours)	MTBF 35°C (in hours)	MTBF 25°C (in hours)	Confidence Level
−1	3500	4200	7385	60%
−2, −3	2000	4680	8190	60%
−4	3156	5100	8925	60%
−17	1960	2875	5031	60%

DESCRIPTION OF THE SYSTEM 310

Intel's System 310 microcomputer (Figure 1) is a user-configurable, MULTIBUS®-based, 16-bit microcomputer intended for real-time applications and interactive multiuser commercial and technical environments. Various configurations are available to meet the customer's optimal price and performance requirements. These configurations are presented in Table 1.

Four key elements of the open system give it power and flexibility:

- iSBC® 86/30 or 286/10 processor boards
- MULTIBUS interface
- XENIX* and iRMX™ operating systems
- Diagnostics

Processor Boards: The heart of the System 310 is the processor board. For the SYP310-1, -2, and -3 the iSBC 86/30 offers the system 128K-byte, dual-port dynamic RAMs, which may be doubled to 256K bytes by adding the iSBC 304C MULTIMODULE™ board, as in the -2 configuration. The -3 version contains an iSBC 012B memory board, extending RAM to 640K bytes. The iSBC 337 MULTIMODULE Math Coprocessor provides arithmetic and logical instruction extensions to the 8086 CPU. The iSBC 337 is optional in the -1 configuration and is standard in the -2 and -3 configurations.

For the SYP310-4 and -17, the iSBC 286/10 provides sites for EPROM, which contains bootstrap and diagnostic code. The 012CX is a 512K-byte RAM board with error correction, connected to the 286/10 on the local bus extension (iLBX™). The iLBX bus provides improved performance by avoiding MULTIBUS arbitration delays. All memory addressable by the processor can be accessed over the iLBX bus (16 M-bytes) and appears to the processor as though it were resident on the processor board.

The MULTIBUS Interface: As industry bus standard IEEE 796, it provides the physical framework and conceptual foundation of Intel's total open system architecture. With a guarantee of future system expansion compatibility, the customer can choose from a wide range of MULTIBUS products. This general-purpose system bus, in conjunction with the iSBC 86/30 or 286/10 processor board, provides a flexible mechanism for intermodule processing, control, and communication. The MULTIBUS interface supports modular memory and I/O expansion. When new peripherals are added, more processing power can be applied to handle them without degrading

* XENIX is a trademark of MicroSoft Corp.

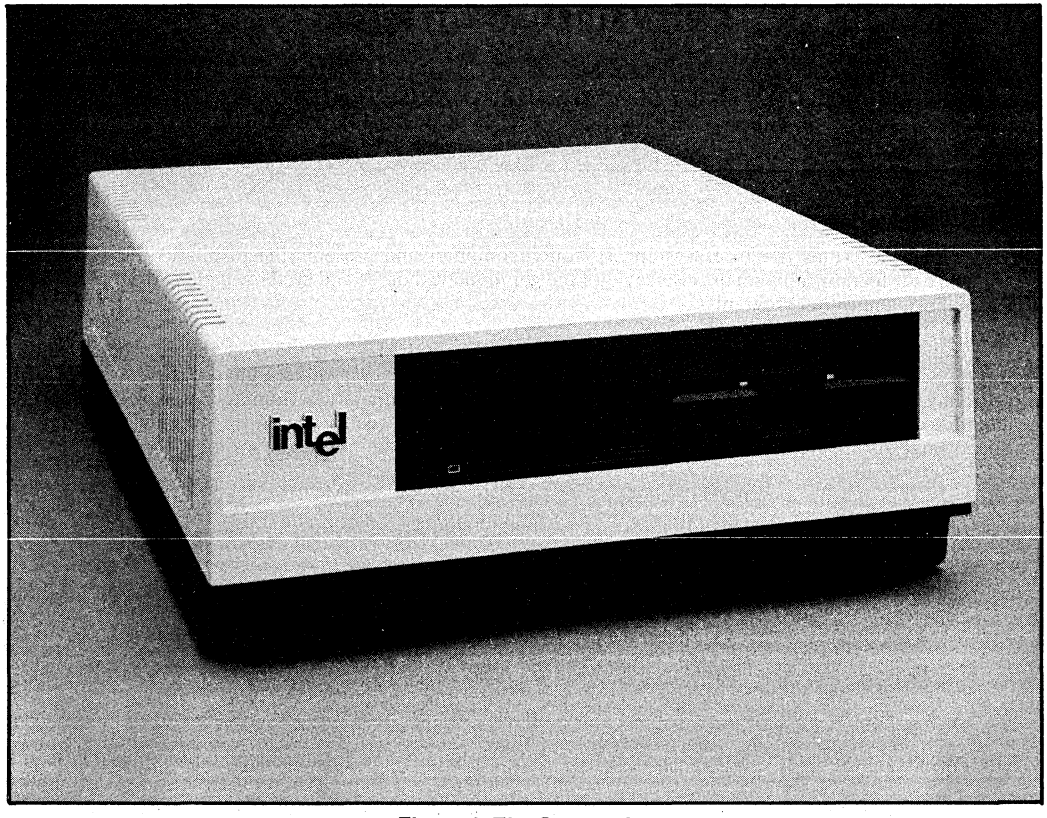


Figure 1. The System 310

Table 1. System 310 Configurations

	-1	-2	-3	-4	-17
iSBC® 86/30 CPU	x	x	x		
iSBC 286/10 CPU				x	x
iSBC 337 math coprocessor		x	x		
Numeric data processor (80287)				x	x
iSBC 304C memory board		x			
iSBC 012B RAM board			x		
iSBC 012CX RAM board				x	x
Flexible diskette drive	x	x	x	x	x
iSBX 218A flexible diskette controller	x	x	x	x	x
Winchester drive (12M-byte)		x	x		
(19M-byte)		x	x		x
iSBC 215G Winchester controller		x	x		x
iSBC 213 Data Separator		x	x		x
System 310 Chassis (including 220W power supply)	x	x	x	x	x

existing processor performance. Figure 2 is a block diagram of the System 310, illustrating the MULTIBUS interface.

Operating Systems: The System 310 operates with the industry-standard iRMX and XENIX operating systems. The iRMX operating system optimizes the System 310 for real-time multitasking, time-critical applications where rapid response is required. XENIX offers conversational access to the System 310 for multiple users in a time-shared environment. A wide range of popular high-level languages enables applications developers to program in multiple languages and still link modules together. The system also supports the Universal Development Interface (UDI), which provides the Independent Software Vendor (ISV) access to the iRMX operating system for developing languages and applications.

Diagnostics: The System 310 includes a comprehensive diagnostic package. A System Confidence Test (SCT) automatically checks the hardware on power-up or RESET. Two separate software packages are also available: the System Diagnostics Tests (SDTs) and the System Analysis Tests (SATs). The SDTs provide detailed diagnostics on problems identified by the SCT and allow the user to isolate a problem on a specific board or drive - often to the component level. The SATs for the iRMX and XENIX operating systems allow the user to interactively exercise the system hardware with the operating system for extended periods of time. Thus, subtle problems can be isolated to a given area within the system.

SYSTEM PRODUCT QUALITY ASSURANCE

The quality and reliability of system products like the System 310 are a function of the quality of components used, the care taken in board and system design and fabrication, and the extent of testing performed on the product before shipment. Examining these functions illustrates the Intel Quality Assurance program for microcomputer system products.

Component Quality Assurance

Standard Intel Component Quality Assurance processing and 100% screening are applied to all Intel components before they are assembled on the boards. Once a component has been qualified as reliable and transferred to systems-level

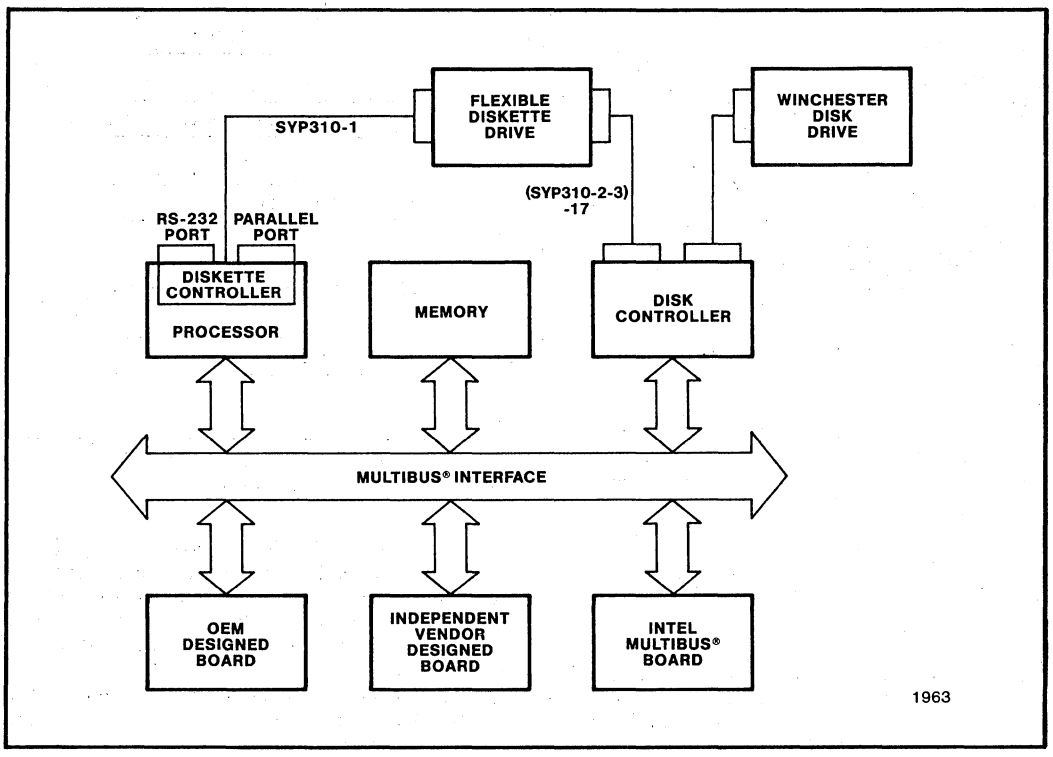


Figure 2. System 310 MULTIBUS® Interface

manufacturing, complete process controls assure the continuation of high quality.

Intel's Component Quality Assurance flow, shown in Figure 3, combines a series of acceptance gates between process steps and detailed inspection at critical points within the processing areas. For example, during wafer fabrication, processes are routinely monitored for contamination through the use of capacitance versus voltage measurements on test chips. Electrical tests, such as breakdown voltage measurements, are performed on test patterns on each wafer. Routine electron microscope examinations at critical process steps also provide important process control feedback. Full functional testing of all parts precedes final Quality Assurance acceptance. Qualified components are then sent to assembly locations.

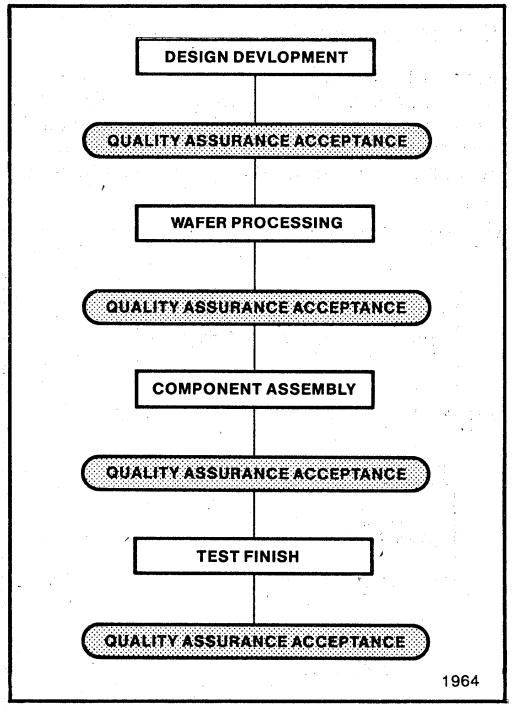


Figure 3. Component Quality Assurance Flow

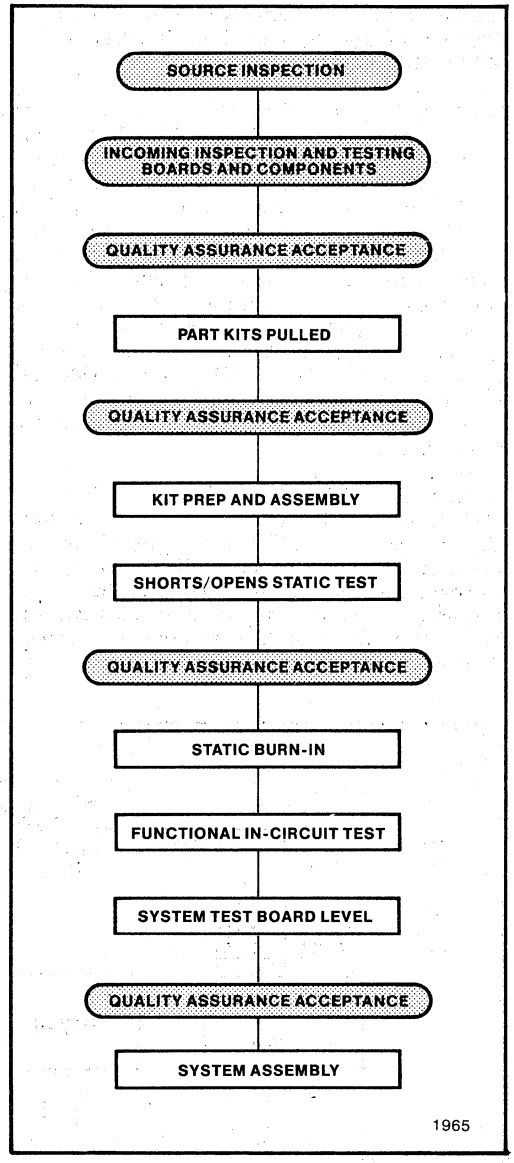


Figure 4. Board Manufacturing, Testing, and Inspection

Board Manufacturing, Testing, and Inspection

The board manufacturing process is closely monitored by Quality Assurance, and inspection occurs at every stage (Figure 4). At Incoming Inspection, bare boards are inspected for board quality. After Incoming Inspection and test, components for an assembly "kit" are then pulled together and readied for the assembly operation. Each kit is visually inspected and then assembled onto the bare board. The assembled parts of the board are inspected prior to wave solder for proper location and after wave solder for soldering defects.

Intel is proud of its wave solder process control. The post-solder cleaning process includes detergent and rinse baths monitored and controlled for concentration, temperature, and contamination.

Boards are selected at random and then checked under a microscope to assure compliance with Intel's wave solder process and cleanliness standards. Boards are also sample tested in an Omegameter (Figure 5), which measures ionic contamination. In the rare event any deviation occurs, the Quality Assurance Inspector stops the process until compliance is restored.

In touch-up and second assembly such items as connectors and process sensitive components are added to a board. As the final phase of kit assembly, a board is tested for circuit "shorts" and "opens" on the Teradyne L529 Automatic Test Equipment (Figure 6). Final assembly inspection certifies the boards are acceptable for test.

The next stage is static burn-in at elevated temperature and voltage margins (70°C, +6% VDC). Burn-in eliminates temperature- and voltage-related "infant mortality" failures that occur early in the life of a product.

The boards are then tested on a Teradyne L125/L135 In-Circuit Tester. This system has a "test the tester" feature that verifies all test points are functioning before actual boards are tested. Use of a special "bed of nails" vacuum fixture ensures that all critical board test points may be accessed by the test system. Any components that fail are replaced, and the boards are retested on the Teradyne system.

Next, boards are functionally tested at System Test Board Level (STBL), a three-stage test. The first stage is at room temperature and nominal voltage. The next two stages are at elevated temperature (70°C) and voltage: first at +6% VDC and then repeated at -6% VDC. For these tests a set of monitor and diagnostic EPROMs are installed, the board is installed into a General Purpose Test Fixture (GPTF) and the board's performance is monitored with a CRT terminal (Figure 7). The monitor/diagnostic exerciser program tests the CPU, I/O interface, and MULTIBUS interface in a multiprocessing environment, which guarantees MULTIBUS interface integrity.

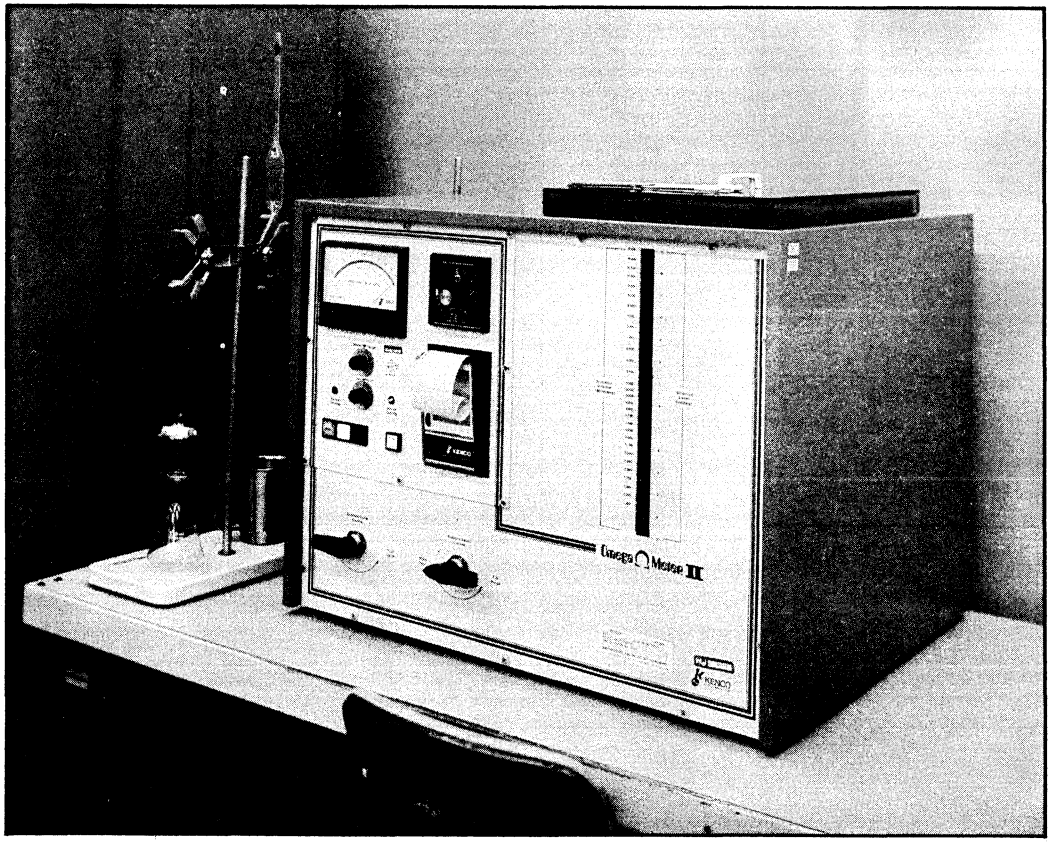


Figure 5. Omegameter: Wave Solder Process Control

Table 2. Environmental Qualification Data

THIS INFORMATION, BASED ON SMALL SAMPLES, IS INTERNAL QUALIFICATION TEST DATA GATHERED UNDER CONDITIONS THAT EXCEED SPECIFICATIONS AND SHOULD NOT BE CONFUSED WITH WARRANTED SPECIFICATIONS. ALL CONFIGURATIONS TESTED PASSED ALL TESTS.

Configuration(Sample)	Test	Description
-1(1) -2(1) -3(4) -4(1) -17(1)	Temperature Map (fully loaded) PASSED	Maximum temperature delta above external system ambient: Semiconductor case temperature +24.4°C (8089 on iSBC 215G) Power supply +19.6°C Internal Winchester +4.5°C
-3(2) -17(1)	Humidity (functional) PASSED (nonfunctional) PASSED	63 hours/unit 29°C (84.2°F) 76% RH noncondensing 90 hours/unit, 26°C wet bulb 95% RH noncondensing
-1(2) -2(2) -3(3) -4(1) -17(1)	ESD PASSED	Direct Discharge Conducted Susceptibility Radiated Susceptibility
-1(2) -2(2) -3(2) -4(1) -17(1)	RFI/EMI PASSED	FCC 47CFR Part 15, "J" for Class A computing devices; VDE 0871, Class A limits
-1(2) -2(2) -3(2) -4(1) -17(1)	Safety PASSED	Listed: UL478 Certified: CSA 22.2 No. 154 IEC435 or VDE0806
-1(2) -2(2) -3(2) -4(1) -17(1)	Voltage Map PASSED	Maximum voltage drop: 100 mv for all supplies at full load at component lead
-1(2) -2(2) -3(4) -4(1) -17(1)	Vibration (nonfunctional) PASSED	5 slews 0-55-0 Hz x,y,z axis 0.01" PTP displacement with 3-minute dwells Axis Resonance x 30 Hz y 48 Hz z 29 Hz

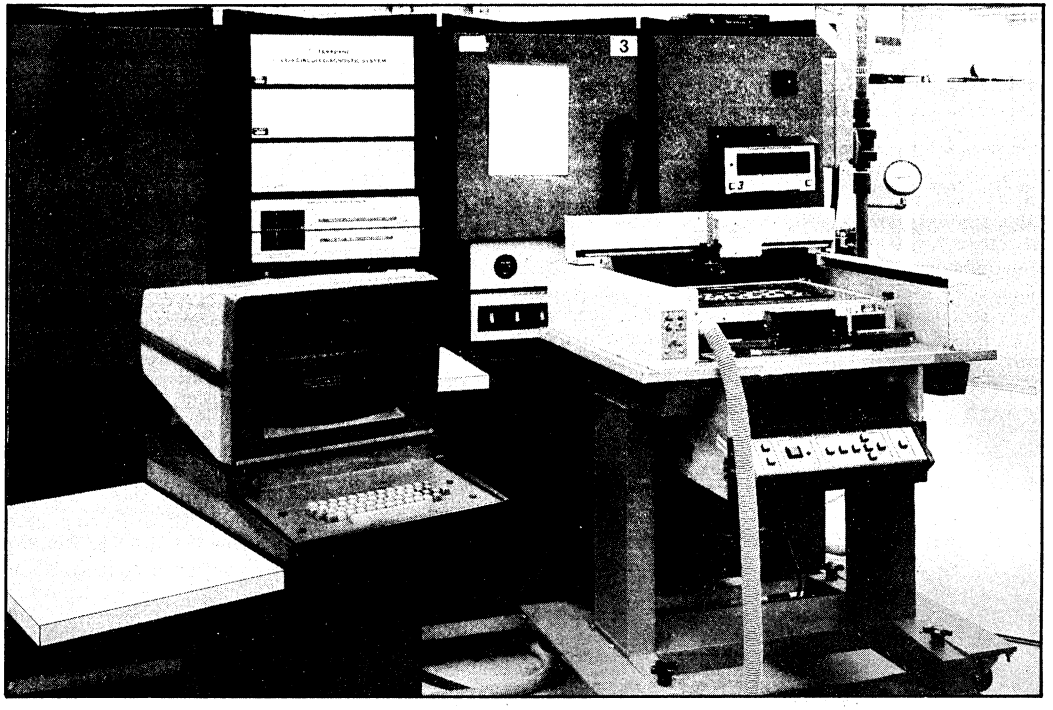


Figure 6. Teradyne Tester

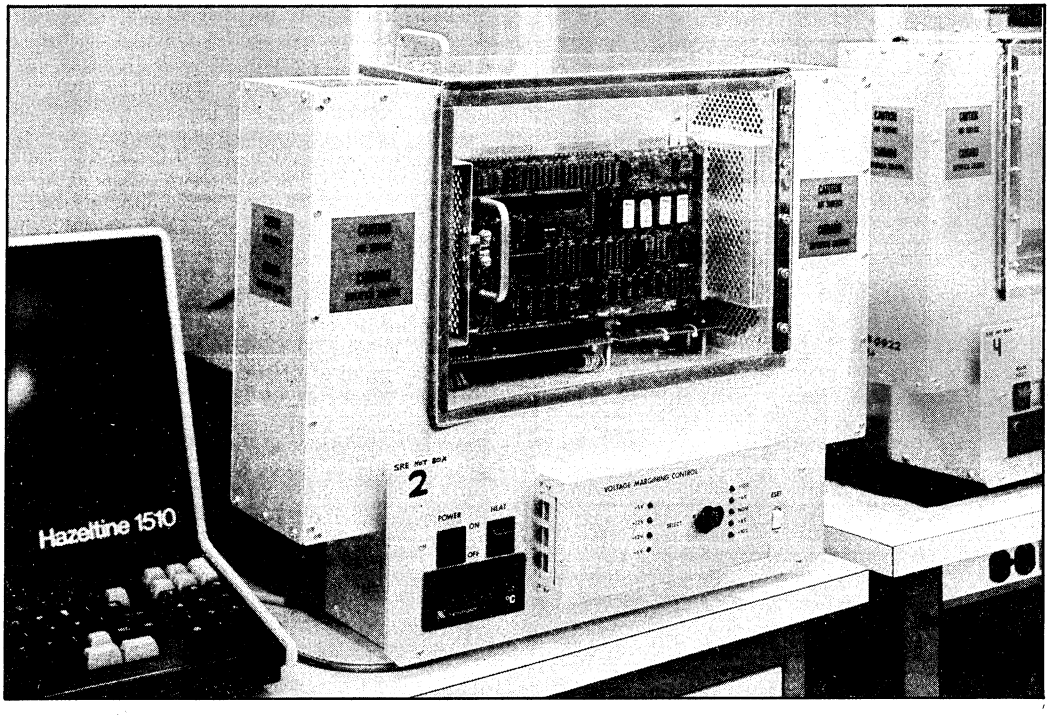


Figure 7. General Purpose Test Fixture

After test, the boards are inspected to ensure that they comply with required specifications and quality standards. Boards passing this final screening are then released by Intel Quality Assurance for use in a system product like the System 310.

System Manufacturing, Testing, and Inspection

Quality Assurance closely monitors the system manufacturing process (Figure 8). After electromechanical assembly, Quality Assurance inspects the systems to ensure compliance with engineering documentation and Intel Workmanship Standards.

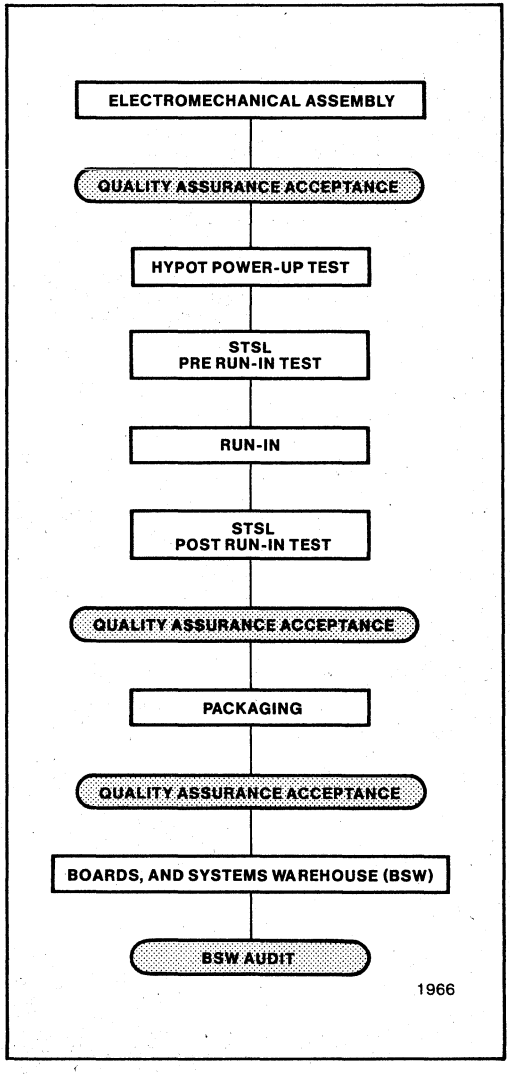


Figure 8. System Manufacturing, Testing, and Inspection

The HYPOT (Hypotronics) test verifies primary circuit integrity and ensures compliance with various safety agency requirements. Next, a power-up test, with load boards, checks power supply output voltages and system power distribution integrity. After the system passes the power-up test, the required boards are installed in the system.

Pre run-in System Test System Level (STSL), a "handshake" test using the System Confidence Test (SCT), is the first test of system integration. This test determines if any major components of the system are malfunctioning. If a problem occurs, it is isolated and corrected, and the system is retested.

STSL, the run-in phase for systems, is functionally equivalent to burn-in for boards. During run-in the system is tested to verify that infant mortality failures have been eliminated. Although the boards have already been rigorously tested, run-in verifies the reliability of the integrated system, which includes power supplies and disk drives. The systems are tested at specification temperature and voltage and are continuously monitored to record any malfunctions. Run-in time is determined by system reliability characterization during the product's development stage. Data collected during the run-in of all subsequent systems are continuously reviewed to assure appropriate run-in time.

At post run-in STSL, Quality Assurance performs a final inspection to certify that all quality standards have been met. After the system passes this inspection, the diskette drive heads are cleaned. As a final step before packaging, drive heads are repositioned to avoid shipping damage.

After packaging, a system is sent to the Boards and Systems Warehouse (BSW) to await shipment to the customer. Quality Assurance conducts monthly finished goods audits on products before shipping. This audit reviews documentation, tests for compliance to environmental specifications (temperature, humidity, shock, vibration), and confirms functional performance over time. Audit systems are run for 48 hours at 35°C. This finished goods audit is a final in-house monitor of the quality that reaches the customer.

Product performance in the field is also tracked. All processes are monitored on a computerized data system, and feedback is used to assure corrective actions when required. This rigorous testing, tracking, and corrective action system assures Intel's product specifications are met, and therefore, the customer's quality and reliability expectations are met.

SYSTEM PRODUCT RELIABILITY

The Life Curve

Three categories of failures can occur during system product life:

- Infant Mortality
- Random
- Wear Out

Each category has a distinct distribution when failure frequency is plotted against time. When the three distributions are combined, the resulting failure rate/time distribution produces a characteristic curve known as a "bathtub" curve. The three distributions, as well as the combined bathtub curve, are represented in Figure 9.

However, the boundaries between the categories are less precise than they appear because the failure categories have overlapping distributions. For example, Infant Mortality failures may extend into the expected Random failure period, but at a very low level. Wear Out failures may, in fact, occasionally occur before the expected Wear Out period. As additional long-term product information becomes available, these boundaries may change.

The Infant Mortality area of the curve shows failures caused by manufacturing defects in the components, boards, and systems. These are "quality failures," and their frequency decreases with time. Infant Mortality for systems depends on the quality and test history of the components and assemblies used in systems manufacturing.

System run-in characterization explores failure modes to identify the change from Infant Mortality to Random failure patterns. The systems are run and failure data collected. Run-in is then defined to include the point at which the instantaneous failure rate falls below a defined limit. Thus, the length of time required for system run-in is closely tied to the Infant Mortality failure rate. In this way, the length of run-in time is defined for future System 310 builds. This point becomes the boundary between the Infant Mortality and Random failure modes.

Random failures occur during the "useful life" of the product, between Infant Mortality and Wear Out. These stress-related failures are primarily a function of temperature, circuit complexity, and device loading. The early phase of the Random failure period is explored during run-in characterization. As the failure pattern becomes random, it approaches a low constant value (flat distribution). For Intel systems this period lasts for many thousands of operating hours. The limiting factors are usually the power supplies, fans, and disk drives, which tend to have higher electrical stress levels and mechanical failures.

To gain insight into system performance, data collected during run-in characterization is extensive. During the original System 310 characterization, the data indicated that run-in should be 40 hours to eliminate Infant Mortality failures. On the

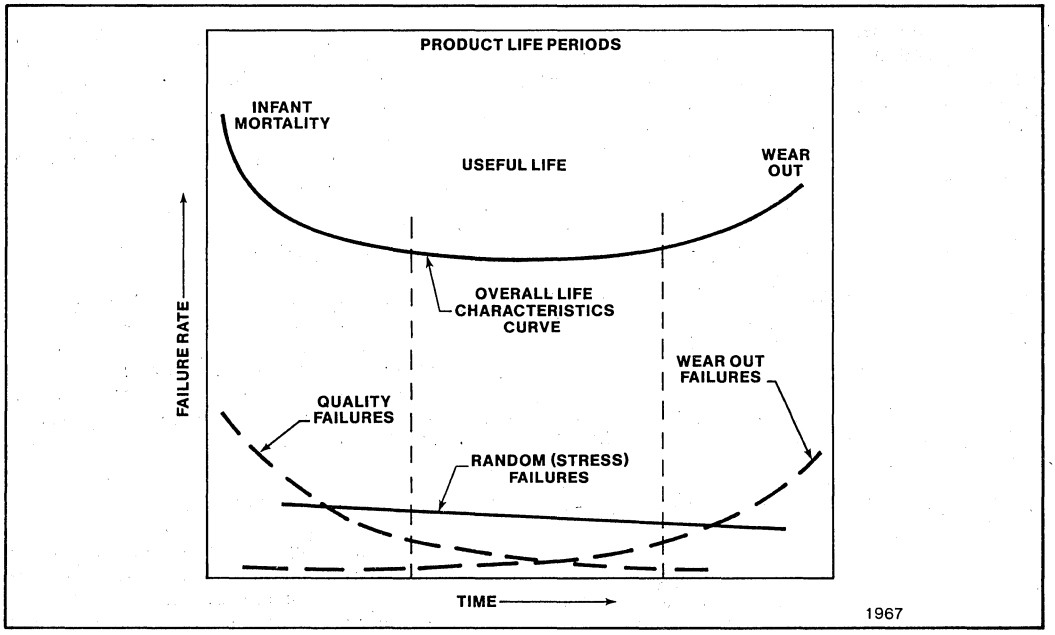


Figure 9. Reliability Life Curve

production floor, after completing the 40 hours of run-in time, we used 145 System 310's to accumulate an additional 19,140 system operating hours with two failures[1], at approximately 25°C. This resulted in a demonstrated 6100 hour MTBF (Mean Time Between Failures; see explanation next section) at 60% confidence level using a Chi-Square distribution[2], and a 3610 hour MTBF at 90% confidence level. This characterization data validated the run-in time for the System 310 and provided some early insights into the Random failure mode.

While run-in defines the starting boundary of useful product life, the end boundary is less easily defined. Until accelerated life testing is completed, the Wear Out period must be estimated, based on known device limitations. As can be seen in Figure 9, Wear Out failure frequency increases with time.

Wear Out failures are primarily due to mechanical wear or chemical degradation. Each can cause the system to lose conformance to specifications. Obviously, not all parts of a system have identical life expectancies. Statistically, Wear Out will not happen until hundreds of years have elapsed for VLSI-based products; however, the life expectancy for peripherals and mechanical devices, such as fans, is in the tens of thousands of hours. Intel identifies these system elements and puts preventive maintenance programs in place.

Mean Time Between Failures and Confidence Levels

In reliability evaluation MTBF is the average time in hours expected to elapse between failures. The point estimate of the MTBF is calculated by dividing the total test operating hours for a sample of system products by the total number of failures during the test period. The demonstrated value at a specific confidence level is calculated using the Chi-Square probability distribution. The confidence level defines the probability that the true MTBF of the product exceeds the demonstrated MTBF value. Because of time constraints on system testing, the required minimum confidence level defined in the reliability test is 60%. This means that 60% of the time the true product MTBF will be higher than the demonstrated MTBF.

As the ratio of operational test hours to the true MTBF increases, the confidence that the true MTBF exceeds the specified lower limit also increases. For example, to demonstrate an MTBF of 5000 hours with a confidence level of 60% takes 10,000 hours of operation with one failure. To demonstrate the same MTBF at a 90% confidence level requires 19,500 hours. The values of MTBF for specific system configurations are normally at the 60% confidence level when the product is qualified for full production. The higher confidence level is attained later in product life, with data from the accelerated life test. As this information becomes available, it will be reported.

RELIABILITY QUALIFICATION

Reliability evaluation serves at least three important functions:

- Demonstrates the MTBF
- Checks the manufacturing process
- Provides system design feedback

First, the evaluation demonstrates the MTBF of a product. This information helps customers to evaluate Intel systems, and helps Intel anticipate customers' product support requirements. Second, it serves as a manufacturing process check. When failures attributed to the manufacturing process occur, they can be quickly identified as exceptions and corrective actions can be implemented. Finally, reliability evaluation serves as a feedback mechanism for system design. Lessons learned from building one system can be used to improve the design of new products. Information gathered from reliability evaluation helps in each of these areas, ensuring Intel will continue to produce the most reliable products possible.

Intel's System Quality and Reliability Department performed a multiphased reliability qualification on the System 310 that included environmental and system reliability testing. Although all boards had already passed extensive board reliability qualifications, the system qualification is also rigorous and complete.

Environmental Testing

A series of environmental tests were performed on the System 310. These tests were designed to assure that the system could withstand the worst conceivable physical conditions that might be found in a commercial or light industrial environment. The environmental tests are described below, and results are summarized in Table 2.

TEMPERATURE MAPPING

Temperature mapping, a study of thermal regions within the environment of the chassis, identifies any excessively high temperatures in a subassembly. Elevated temperatures can cause decreased life expectancy and lower system reliability. All

1 Both failures were Winchester controllers: one at 130 operating hours, the other at 141 operating hours. This problem has been resolved.

2 See Reference 1.

Table 2. Environmental Qualification Data (continued)

Configuration(Sample)	Test	Description
-1(2) -3(2) -4(1) -17(1)	Vibration (functional) PASSED	7-55 Hz, in 1 Hz steps 1 minute dwell per step 30 minute dwell at the resonant frequency or 55 Hz Axis / Resonance / PTP x / 49 Hz / .0045" y,z / 55 Hz / .0045" x / 44 Hz / .003" y / 48 Hz / .003" z / 54 Hz / .001"
-3(3)	Shock (nonfunctional) PASSED	15G, 1 drop per face (6)
-3(2)	Thermal Shock (non-functional) PASSED	8 cycles 60°C to −40°C
-3(1)	Altitude (functional) PASSED	Altitude chamber: 8000 feet 35°C, 220W maximum load
-3(5)	Packaging PASSED	Tested to National Safe Transit Requirements
-3(1)	Package Drop PASSED	Republic Packaging: 30", 15G maximum, 6 faces
-3(12)	Shipping PASSED	Inspection and functional test before and after round trip shipment: Portland, OR - San Jose, CA (plane change at San Jose)

measurements were taken in a walk-in environmental chamber at 35°C (Figure 10). The iRMX 86 System Analysis Test (RSAT) exercised the system during temperature mapping. Measurements on all subassemblies within the chassis (boards, drives, and power supply) were well within their individual maximum operating temperature specifications.

HUMIDITY

The humidity test identifies problems related to high temperature and high humidity environments, such as corrosion and poor mechanical contacts. These conditions not only test the reliability of the systems at high humidity and temperature, but also accelerate the effects of moderate humidity over time. The humidity test confirmed that the system operates properly at its maximum 26°C wet bulb specification. No evidence of system deterioration was found at the end of the test.

Note, however, that the system may not function correctly if operated at its maximum temperature specification (35°C) and its maximum humidity specification (80% relative humidity) simultaneously. These conditions fall outside the maximum 26°C wet bulb specification (a disk drive limitation) and the defined operating environment of expected system use. The temperature and humidity boundaries for 26°C wet bulb are illustrated in Figure 11.

EMI, RFI, ESD

Electro Magnetic Interference (EMI) and Radio Frequency Interference (RFI) evaluation measures the emitted and conducted radiation from the product to the outside world. Electro Static Discharge (ESD) evaluation measures a system's ability to withstand static discharges of electricity without failure or component damage. An independent laboratory confirmed that the System 310 passes FCC (Federal Communications Commission) and VDE (Verband Deutscher Elektrotechniker) conducted and radiated emission requirements. ESD specifications were verified internally.

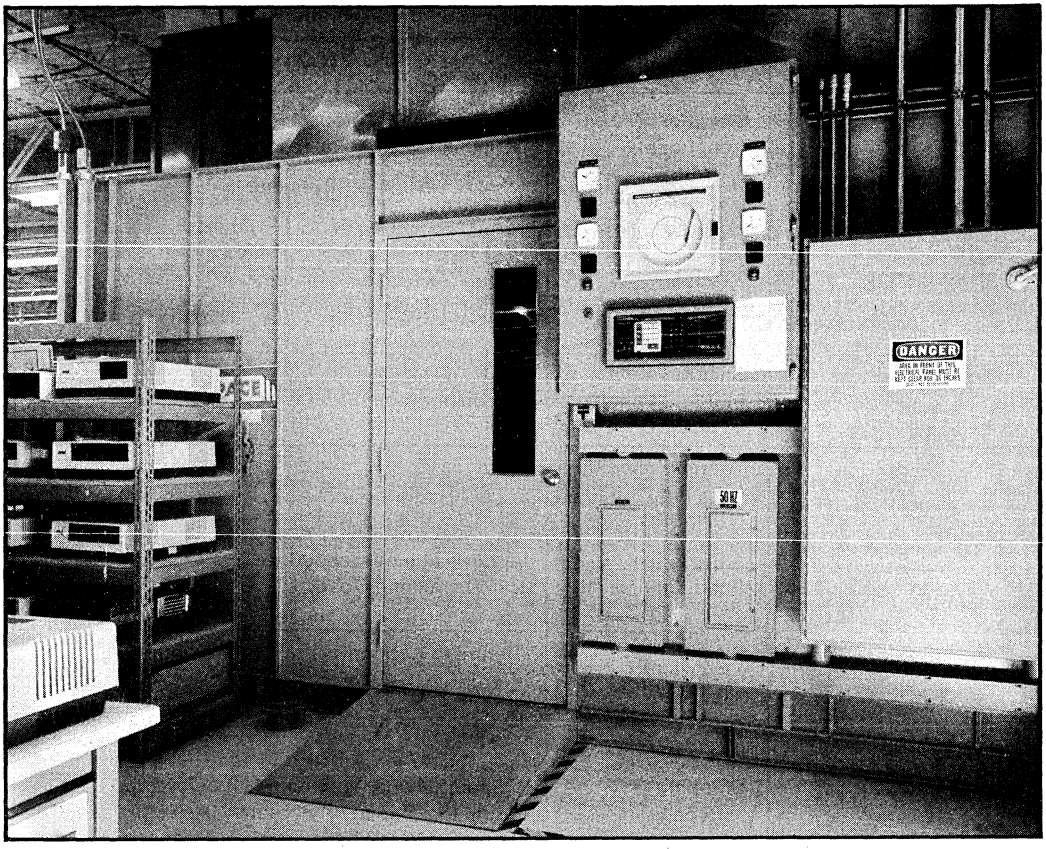


Figure 10. Environmental Chamber

VOLTAGE MAPPING

Voltage mapping measures the voltage potential at various points within the system and ensures no excessive voltage drops exist in the system. All points measured were well within the tolerances allowed for operation.

VIBRATION AND SHOCK

Vibration and shock testing identifies mechanically related problems a system could sustain under normal conditions. The diskette and Winchester drives' vibration and shock specifications are lower than those of the remainder of the system (boards, fans, power supply, chassis structure); the system limits are essentially those of the peripherals. Functional and non-functional vibration tests were performed both at Intel and in an independent laboratory. Non-functional shock tests were performed at two independent facilities.

The test system passed the functional and non-functional vibration tests from 5Hz to 55Hz. As expected, the critical point in the system was the Winchester drive head assembly. No damage resulted, nor did the tests have any permanent effect on the performance of the system.

ALTITUDE

The altitude test evaluates the system's performance at altitude and temperature extremes. The system was operated at maximum altitude (8000 ft.) and maximum temperature (35°C) with maximum power supply loading (220W) without exceeding the temperature specifications of the various components.

PACKAGING AND SHIPPING

System packaging requirements are rigorously tested under both laboratory and actual shipping conditions. In the laboratory, packaged systems were subjected to worst-case conditions. Systems were monitored for shock, and the package was accepted only after shock to the system met the specification.

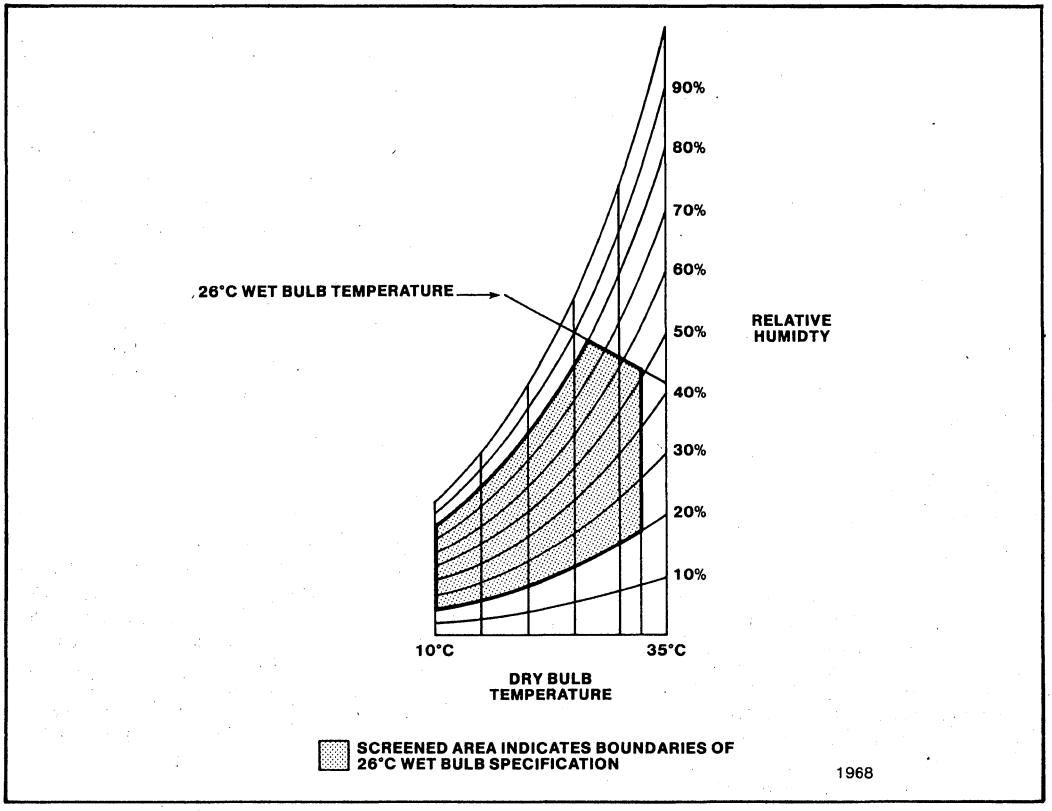


Figure 11. Operational Temperature and Humidity Specification, 26°C Wet Bulb

The reliability evaluation confirms the adequacy of packaging for commercial transportation conditions. Twelve systems were shipped round trip by truck and air from Hillsboro, Oregon, to Santa Clara, California, (approximately 1600 miles). During this test they were subjected to the typical handling conditions of commercial carriers. Before and after the test the systems were visually inspected and mechanically tested; no damage resulted.

The results of these environmental tests are summarized in Table 2.

Reliability Test Evaluation

Reliability test data were collected on System 310 configurations:

- 310-1
- 310-2, -3
- 310-4
- 310-17

Because of differences in system complexity and the Winchester drive in the 310-2, -3, and -17, each configuration had a different minimum MTBF qualification goal.

All System 310 reliability evaluations surpassed their designated minimum MTBF qualifications. The qualification requirements were at 35°C with a confidence level of 60%. The minimum MTBF requirements and the actual MTBFs attained are summarized in Table 3.

The elevated temperature (35°C) during the evaluation accelerates the wear on a system. Although the MTBF is demonstrated at 35°C, an MTBF at 25°C can be derived from the test data using a factor of 1.75 based on the Arrhenius equation[1]. This information is also presented.

Two types of tests are performed: the initial reliability qualification test and the accelerated life test. The initial reliability test qualifies a product for full production. This test is time truncated; the test is terminated when the product has met its minimum required MTBF. At that time, the product is qualified for full production.

[1] See Reference 2.

The accelerated life test is designed to establish the unit's actual MTBF and to gather information on a product's long term reliability. This test begins after a product has passed its initial reliability qualification test. It continues until the end of life of the unit. The System 310 accelerated life tests are still in progress.

SYSTEM 310-1

Initial Qualification Test: For the System 310-1 configuration, the reliability qualification required a minimum demonstrated MTBF of 3500 hours, 60% confidence level, at 35°C. Ten units were evaluated. The test was terminated after accumulating 3870 hours with no failures. The test yielded an MTBF of 4200 hours at 60% confidence level (35°C) qualifying it for full production. The derived MTBF at 25°C is 7385 hours at 60% confidence level. No Accelerated Life Test is being run on the 310-1 configuration.

SYSTEM 310-2 AND -3

Three reliability tests were conducted on -2 and -3 configurations combined. The first two tests were time truncated. One test resulted in no failures; the second test had three failures at termination. The third test, an accelerated life test, has yielded no failures to date. Although this evaluation is still in progress, the data accumulated to date (July 1, 1984) are reported. As the data increases, so does our understanding of the reliability of the system. The accelerated life test data will continue to accumulate, and they will be reported.

Initial Qualification Test: For the System 310-2 and -3 configurations, the reliability qualification required a minimum demonstrated MTBF of 2000 hours, 60% confidence level, at 35°C. Twenty-five units were evaluated. The test was terminated after accumulating 1870 system hours with no failures. The test met an MTBF of 2020 hours at 60% confidence level, and this product was qualified for full production. The derived MTBF at 25°C is 3552 hours at 60% confidence level.

The complexity of the -2 and -3 system configurations required more complete evaluation. Failure patterns of complex systems are largely unknown, so two further tests were designed: one to be terminated at about10,000 hours; the other an accelerated life test, still in progress. The 10K-hour test logged 10,750 system hours at 35°C with three failures. Twenty units were evaluated. This was a rigorous test with the systems fully loaded to simulate maximum power load conditions. At 35°C this test yielded a 2570-hour MTBF at 60% confidence level. The 25°C derived MTBF is 4519 hours at 60% confidence level.

Accelerated Life Test: This test, still in progress, will measure the end of life of the system. Two fully loaded systems are running at elevated temperature twenty-four hours a day. As of July 1, 1984, the two systems had accumulated 7166 operating hours with no failures. This yields an MTBF of 7790 hours at 35°C, 60% confidence level and a derived MTBF of 13,632 hours at 25°C, 60% confidence level. These MTBFs should be interpreted with caution because the sample size is small and no failures have occurred.

Total System 310-2, -3 Test Experience: Forty-seven System 310-2 and -3 units have been tested, accumulating 19,786 system operating hours. The MTBF at 35°C is 4680 hours at 60% confidence level; 2900 hours at 90% confidence level. At 25°C, the derived MTBF is 8190 hours at 60% confidence level; 5075 hours at 90% confidence level.

SYSTEM 310-4

Initial Qualification Test: For the System 310-4 (80286-based) configuration, the reliability qualification required a minimum demonstrated MTBF of 3156 hours at 60% confidence level, 35°C. Fourteen units were evaluated. The test was terminated after 13,992 hours with two failures. The test yielded an MTBF of 4400 hours at 60% confidence level, 35°C, thus qualifying the product for full production. The derived MTBF at 25°C is 7700 hours at 60% confidence level.

Accelerated Life Test: This test was started on two 310-4 units. These units are running twenty-four hours a day at 35°C. As of July 1, 1984, the two systems had accumulated 2021 operating hours with no failures. The MTBF is 2200 hours at 35°C, 60% confidence level; the derived MTBF at 25°C is 3850 hours, 60% confidence level.

Total System 310-4 Test Experience: A total of 16,013 operating hours with two failures have been accumulated on the 16 System 310-4 units. This yields an MTBF of 5100 hours at 35°C, 60% confidence level; 3000 hours at 90% confidence level. The derived MTBF at 25°C is 8925 hours at 60% confidence level; 5250 hours at 90% confidence level.

SYSTEM 310-17

Initial Qualification Test: The System 310-17 qualification required a minimum demonstrated MTBF of 1960 hours at 60% confidence level, 35°C. Thirteen units were evaluated. The test was terminated after 13,434 hours with four failures. The test yielded an MTBF of 2500 hours at 60% confidence level, 35°C. The derived MTBF at 25°C was 4375 at 60% confidence level.

Accelerated Life Test: This test is still in progress with two 310-17 units; conditions are identical to those for the 310-4 units. As of July 1, 1984, 2021 operating hours have been accumulated on the two units with no failures. The MTBF at 35°C is 2200 hours at 60% confidence level; the derived MTBF at 25°C is 3850 hours at 60% confidence level.

Total System 310-17 Test Experience: A total of 15,455 operating hours with four failures have been accumulated on the 15 System 310-17 units. This yields an MTBF of 2875 hours at 35°C, 60% confidence level; 1800 hours at 90% confidence level. The derived MTBF at 25°C is 5031 hours at 60% confidence level; 3150 hours at 90% confidence level.

REFERENCES

1. *Reliability Mathematics: Fundamentals; Practices; and Procedures*, B. L. Amstadler, 1971. McGraw-Hill, Inc.

2. *Semiconductor Device Reliability and the Arrhenius Model*, National Semiconductor Reliability Report G-11, January 1977.

Table 3. Summary of Reliability Tests

-1 CONFIGURATION					
Operating Hours	**Failures**	**MTBF Goal 35°C (in hours)**	**MTBF 35°C (in hours)**	**MTBF 25°C (in hours)**	**Confidence Level**
3,870 (time truncated) n=10	0	3500	4200	7385	60%

-2, -3 CONFIGURATIONS					
Operating Hours	**Failures**	**MTBF Goal 35°C (in hours)**	**MTBF 35°C (in hours)**	**MTBF 25°C (in hours)**	**Confidence Level**
1,870 (time truncated) n=25	0	2000	2020	3,552	60%
10,750 (time truncated) n=20	3*		2570	4,519	60%
7,166 (on-going) n=2	0		7790	13,632	60%
TOTAL FOR -2, -3					
19,786 n=47	3		4680 2900	8190 5075	60% 90%

* Failures include
1) Resettable Winchester control error
2) Winchester control error (marginal crystal oscillator)
3) Power supply failure

Table 3. Summary of Reliability Tests (continued)

-4 CONFIGURATION					
Operating Hours	Failures	MTBF Goal 35°C (in hours)	MTBF 35°C (in hours)	MTBF 25°C (in hours)	Confidence Level
13,992 (time truncated) n=14	2*	3156	4400	7700	60%
2,021 (on-going n=2	0		2200	3850	60%
TOTAL FOR -4					
16,013 n=16	2		5100 3000	8925 5250	60% 90%

* Failures
1) Flexible diskette drive intermittent
2) Response failure; occurred once, no reoccurrence

-17 CONFIGURATION					
Operating Hours	Failures	MTBF Goal 35°C (in hours)	MTBF 35°C (in hours)	MTBF 25°C (in hours)	Confidence Level
13,434 (time truncated) n=13	4*	1960	2500	4375	60%
2,021 (on-going) n=2	0		2200	3850	60%
TOTAL FOR -17					
15,455 n=15	4		2875 1800	5031 3150	60% 90%

* Failures
1) Winchester drive
2) 215G intermittent
3) System halt, user resettable
4) RAM malfunction, user resettable

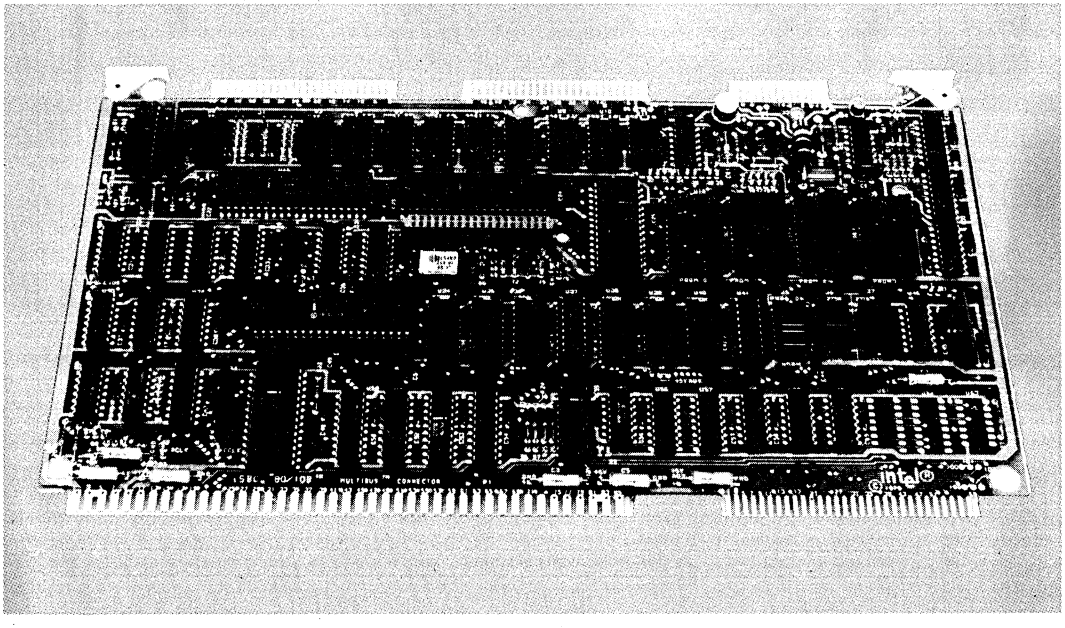
Single Board Computers

2

iSBC® 80/10B (or pSBC 80/10B*) SINGLE BOARD COMPUTER

- 8080A CPU used as central processing unit
- One iSBX™ bus connector for iSBX™ MULTIMODULE™ board expansion
- 1K byte of read/write memory with sockets for expansion up to 4K bytes
- Sockets for up to 16K bytes of read only memory
- 48 programmable parallel I/O lines with sockets for interchangeable line drivers and terminators
- Programmable synchronous/asynchronous communications interface with selectable RS232C or teletypewriter compatibility
- Single level interrupt with 11 interrupt sources
- Auxiliary power bus and power-fail interrupt control logic for RAM battery backup
- 1.04 millisecond interval timer
- Limited master MULTIBUS® interface

The Intel® iSBC 80/10B board is a member of Intel's complete line of OEM microcomputer systems which take full advantage of Intel's LSI technology to provide economical, self-contained computer-based solutions for OEM applications. The iSBC 80/10B board is a complete computer system on a single 6.75 x 12.00-inch printed circuit card. The CPU, system clock, iSBX bus interface, read/write memory, read only memory sockets, I/O ports and drivers, serial communications interface, bus control logic, and drivers all reside on the board.



*Same product, manufactured by Intel Puerto Rico, Inc.

October, 1984
AFN-01688B

FUNCTIONAL DESCRIPTION

Intel's powerful 8-bit n-channel MOS 8080A CPU, fabricated on a single LSI chip, is the central processor for the iSBC 80/10B board. The 8080A contains six 8-bit general purpose registers and an accumulator. The six general purpose registers may be addressed individually or in pairs, providing both single and double precision operators. A block diagram of iSBC 80/10B board functional components is shown in Figure 1.

iSBX™Bus MULTIMODULE™Board Expansion

The new iSBX bus interface brings an entirely new dimension to system design offering incremental on-board expansion with small iSBX boards. One iSBX bus connector interface is provided to accomplish plug-in expansion with any iSBX MULTIMODULE board. iSBX boards are available to provide expansion equivalent to the I/O available on the iSBC 80/10B board or the user may configure entirely new functionality such as math directly on-board. The iSBX 350 programmable I/O MULTIMODULE board provides 24 I/O lines using an 8255A programmable peripheral interface. Therefore, the iSBX 350 module together with the iSBC 80/10B board may offer 72 lines of programmable I/O. Alternately, a serial port may be added using the iSBX 351 serial I/O multimodule board or math may be configured on-board with the iSBX 332 floating point math MULTIMODULE board.

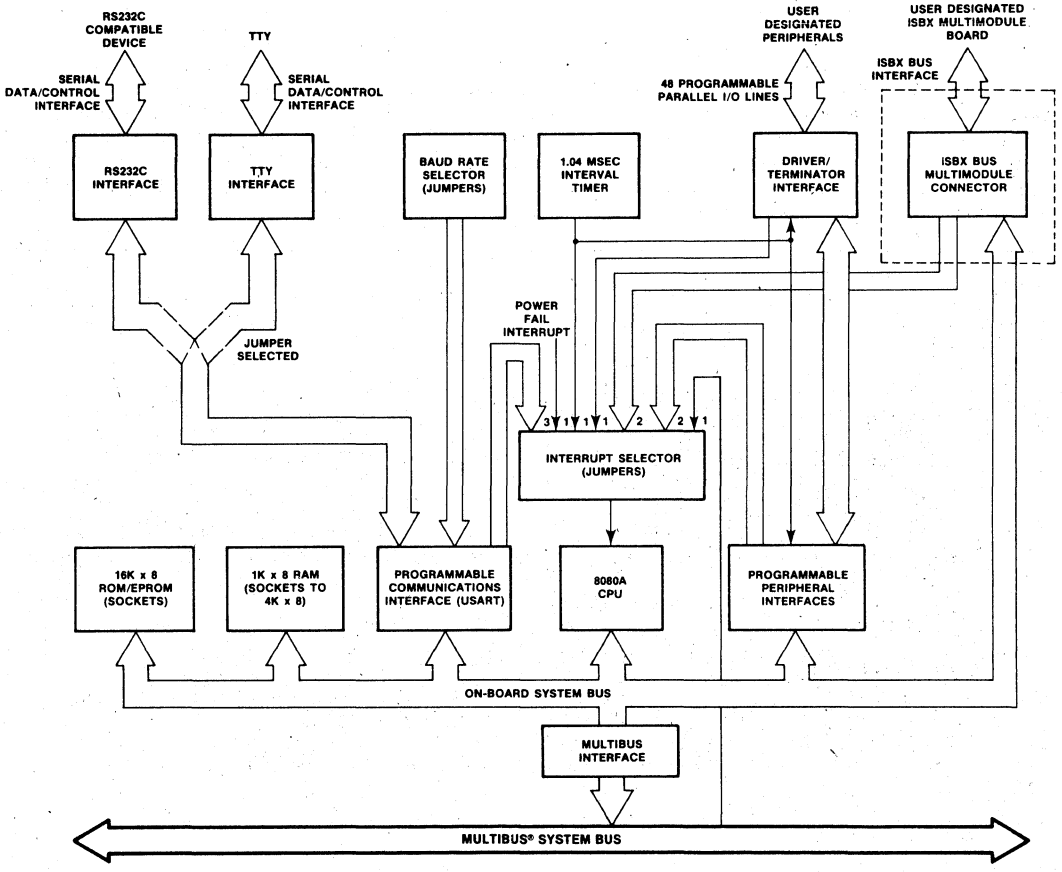


Figure 1. iSBC® 80/10B Single Board Computer Block Diagram

The iSBX board is a logical extension of the on-board programmable I/O and is accessed by the iSBC 80/10B single board computer as common I/O port locations. The iSBX board is coupled directly to the 8080A CPU and therefore becomes an integral element of the iSBC 80/10B single board computer providing optimum performance.

Memory Addressing

The 8080A has a 16-bit program counter which allows direct addressing of up to 64K bytes of memory. An external stack, located within any portion of read/write memory, may be used as a last-in/first-out storage area for the contents of the program counter, flags, accumulator, and all of the six general purpose registers. A 16-bit stack pointer controls the addressing of this external stack. This stack provides subroutine nesting bounded only by memory size.

Memory Capacity

The iSBC 80/10B board contains 1K bytes of read/write static memory. In addition, sockets for up to 4K bytes of RAM memory are provided on board. Read/write memory may be added in 1K byte increments using two 1K×4 Intel 2114A-5 static RAMs. All on-board RAM read and write operations are performed at maximum processor speed. Sockets for up to 16K bytes of nonvolatile read-only-memory are provided on the board. Read-only-memory may be added in 1K byte increments up to 4K bytes (using Intel 2708 or 2758); in 2K byte increments up to 8K bytes (using Intel 2716); or in 4K byte increments up to 16K bytes (using Intel 2732). All on-board ROM or EPROM read operations are performed at maximum processor speed.

Parallel I/O Interface

The iSBC 80/10B board contains 48 programmable parallel I/O lines implemented using two Intel 8255A programmable peripheral interfaces. The system software is used to configure the I/O lines in any combination of unidirectional input/output, and bidirectional ports indicated in Table 1. Therefore, the I/O interface may be customized to meet specific peripheral requirements. In order to take full advantage of the large number of possible I/O configurations, sockets are provided for interchangeable I/O line drivers and terminators. Hence, the flexibility of the I/O interface is further enhanced by the capability of selecting the appropriate combination of optional line drivers and terminators to provide the required sink current, polarity, and drive/termination characteristics for each application. The 48 programmable I/O lines and signal ground lines are brought out to two 50-pin edge connectors that mate with flat cable or round cable.

Serial I/O Interface

A programmable communications interface using the Intel® 8251A Universal Synchronous/Asynchronous Receiver/Transmitter (USART) is contained on the board. A jumper selectable baud rate generator provides the USART with all common communications frequencies. The USART can be

Table 1. Input/Output Port Modes of Operation

Port	Lines (qty)	Mode of Operation					
		Unidirectional				Bidirectional	Control
		Input		Output			
		Unlatched	Latched & Strobed	Latched	Latched & Strobed		
1	8	X	X	X	X	X	
2	8	X	X	X	X		
3	8	X		X			X[1]
4	8	X		X			
5	8	X		X			
6	4	X		X			
	4	X		X			

Notes

Port 3 must be used as a control port when either port 1 or port 2 are used as a latched and strobed input or a latched and strobed output port or port 1 is used as a bidirectional port.

programmed by the system software to select the desired synchronous or asynchronous serial data transmission technique (including IBM Bi-Sync). The mode of operation (i.e., synchronous or asynchronous), data format, control character format and parity are all under program control. The 8251A provides full duplex, double-buffered transmit and receive capability. Parity, overrun, and framing error detection are all incorporated in the USART. The inclusion of jumper selectable TTY or RS232C compatible interfaces on the board, in conjunction with the USART, provides a direct interface to teletypes, CRTs, RS232C compatible cassettes, and asynchronous and synchronous modems. The RS232C or TTY command lines, serial data lines, and signal ground lines are brought out to a 26-pin edge connector that mates with RS232C compatible flat or round cable.

Interrupt Capability

Interrupt requests may originate from 11 sources. Two jumper selectable interrupt requests can be automatically generated by the programmable peripheral interface when a byte of information is ready to be transferred to the CPU (i.e., input buffer is full) or a byte of information has been transferred to a peripheral device (i.e., output buffer is empty). Three jumper selectable interrupt requests can be automatically generated by the USART when a character is ready to be transferred to the CPU (i.e., receive channel buffer is full), a character is ready to be transmitted (i.e., the USART is ready to accept a character from the CPU), or when the transmitter is empty (i.e., the USART has no character to transmit). These five interrupt request lines are all maskable under program control. Two interrupt request lines may be interfaced directly to user designated peripheral devices; one via the MULTIBUS system bus and the other via the I/O edge connector. One jumper selectable interrupt request may be interfaced to the power-fail interrupt control logic. One jumper selectable interrupt request may originated from the interval timer. Two general purpose interrupt requests are jumper selectable from the iSBX interface. These two signals permit a user installed MULTIMODULE board to interrupt the 8080A CPU. The eleven interrupt request lines share a single CPU interrupt level. When an interrupt request is recognized, a restart instruction (RESTART 7) is generated. The processor responds by suspending program execution and executing a user defined interrupt service routine originating at location 38_{16}.

Power-Fail Control

A power-fail interrupt may be detected through the AC-low signal generated by the power supply. This signal may be configured to interrupt the 8080A CPU to initiate an orderly power down instruction sequence.

Interval Timer

A 1.04 millisecond timer is available for interval interrupts or as a clock output to the parallel I/O connector. The timer output is jumper selectable to the programmable parallel interface, the parallel I/O connector (J1), or directly to the 8080A CPU.

MULTIBUS® System Expansion Capabilities

Memory and I/O capacity may be expanded and additional functions added using Intel MULTIBUS™ system compatible expansion boards. Memory may be expanded to 65,536 bytes by adding user specified combinations of RAM boards, EPROM boards, or combination boards. Input/output capacity may be increased by adding digital I/O and analog I/O expansion boards. In addition, the iSBC 80/10B board performs as a limited bus master in that it must occupy the lowest priority when used with other MULTIBUS masters. The bus master may take control of the MULTIBUS system bus by halting the iSBC 80/10B board program execution. Mass storage capability may be achieved by adding single density diskette, double density diskette, or hard disk controllers. Modular expandable backplanes and cardcages are available to support multiboard systems.

System Development Capability

The development cycle of iSBC 80/10B-based products may be significantly reduced using Intel's system development tools available today. For those not requiring hardware emulation capability, Intel provides a new low cost microcomputer development system. The iPDS, Personal Development System, provides low cost system develop-

ment for the iSBC 80/10B board, while at the same time providing personal computer capability for the engineer. The Intellec Series II family of compatible microcomputer development systems provides a range of capability from a low cost disk-based edit debug workstation to a high performance, fully compatible hard-disk-based software development system. A unique in-circuit emulator (ICE-80) option provides the capability of developing and debugging software directly on the iSBC 80/10B.

Programming Capability

PL/M-80 — Intel's high level programming language, PL/M, is also available as a resident Intellec microcomputer development system option. PL/M provides the capability to program in a natural, algorithmic language and eliminates the need to manage register usage or allocate memory. PL/M programs can be written in a much shorter time than assembly language programs for a given application.

FORTRAN-80 — For applications requiring computational and formatted I/O capabilities, the ANSI 77 standard high level FORTRAN-80 programming language is available as a resident option of the Intellec system. The FORTRAN compiler produces relocatable object code that may be easily linked with PL/M or assembly language program modules. In addition, the iSBC 801 FORTRAN-80 run-time package is a complete, ready-to-use set of linkable object modules which are fully compatible with iRMX 80 systems. The modules, when combined with the FORTRAN-80 coded application, provide the appropriate interfaces to the disk file and terminal I/O of iRMX 80, and to the iSBC 310A Math Unit for applications requiring high speed math.

BASIC-80 — A high level language interpreter is available with extended disk capabilities which operates under the iRMX 80 Real-Time Multitasking Executive and translates BASIC-80 source programs into an internally executable form. This language interpreter, provided as a set of linkable object modules, is ideally suited to the OEM who requires a pass through programming language. The BASIC-80 programs may be created, stored, and interpreted on the iSBC 80 based systems using the iSBC 802 BASIC-80 Configurable iRMX 80 Disk-Based Interpreter. The iSBC 802 Interpreter has a complete ready-to-use set of linkable object modules which are fully compatible with Intel's iRMX 80 Real-Time Multitasking Executive Software. The modules provide interfaces to disk file and terminal I/O, software floating point, or interface to other routines provided by the user.

SPECIFICATIONS

Word Size

Instruction — 8, 16, or 24 bits
Data — 8 bits

Cycle Time

Basic Instruction Cycle — 1.95 μsec

Note
Basic instruction cycle is defined as the fastest instruction (i.e., four clock cycles).

Memory Addressing

On-Board ROM/EPROM

0-0FFF using 2708, 2758
0-1FFF using 2716
0-3FFF using 2732

On-Board RAM

3C00-3FFF with no RAM expansion
3000-3FFF with 2114A-5 expansion

Note
All RAM configurations are automatically moved up to a base address of 4XXX when configuring EPROM for 2732.

Memory Capacity

On-Board ROM/EPROM

16K bytes (sockets only)

On-Board RAM

1K byte with user expansion in 1K increments to 4K bytes using Intel 2114A-5 RAMs

Off-Board Expansion

Up to 64K bytes using user specified combinations of RAM, ROM, and EPROM.

I/O Addressing

On-Board Programmable I/O

Device	I/O Address
8255A No. 1	
Port A	E4
Port B	E5
Port C	E6
Control	E7
8255A No. 2	
Port A	E8
Port B	E9
Port C	EA
Control	EB
8251A	
Data	EC
Control	ED
iSBX Multimodule	
MCS0	F0-F7
MCS1	F8-FF

I/O Capacity

Parallel — 48 programmable lines
Serial — 1 transmit, 1 receive
MULTIMODULE — 1 iSBX Bus MULTIMODULE Board

Serial Baud Rates

Frequency (kHz) (Jumper Selectable)	Baud Rate (Hz)		
	Synchronous	Asynchronous (Program Selectable)	
		÷ 16	÷ 64
307.2	—	19200	4800
153.6	—	9600	2400
76.8	—	4800	1200
38.4	38400	2400	600
19.2	19200	1200	300
9.6	9600	600	150
6.98	6980	—	110
4.8	4800	300	75

Serial Communications Characteristics

Synchronous — 5-8 bit characters; internal or external character synchronization; automatic sync insertion

Asynchronous — 5-8 bit characters; break character generation; 1, 1½, or 2 stop bits; false start bit detectors

Interrupts

Single-level with on-board logic that automatically vectors the processor to location 38H using a restart instruction (RESTART7). Interrupt requests may originate from user specified I/O (2); the programmable peripheral interface (2); the iSBX MULTIMODULE board (2); the programmable communications interface (3); the power fail interrupt (1); or the interval timer (1).

Interfaces

MULTIBUS — All signals TTL compatible

iSBX Bus — All signals TTL compatible

Parallel I/O — All signals TTL compatible

Serial I/O — RS232C or a 20 mil current loop TTY interface (jumper selectable)

Interrupt Requests — All TTL compatible (active-low)

Clocks

System Clock — 2.048 MHz ± 0.1%

Interval Timer — 1.042 msec ± 0.1% (959.5 HZ)

Connectors

Interface	Double-Sided Pins (qty)	Centers (in.)	Mating Connectors
MULTIBUS System	86	0.156	Viking 2KH43/9AMK12 Wire-wrap
iSBX Bus	36	0.1	iSBX 960-5
Parallel I/O (2)	50	0.1	3M 3415-000 Flat
Serial I/O	26	0.1	AMP 87194-6 Flat

Physical Characteristics

Width — 12.00 in. (30.48 cm)

Height — 6.75 in. (17.15 cm)

Depth — 0.05 in. (1.27 cm)

Weight — 14 oz. (397.3 gm)

Electrical Characteristics

DC Power Requirements

Voltage	Without EPROM[1]	With 2708 EPROM[2]	With 2758, 2716, or 2732 EPROM[3]	Power Down Requirements (RAM and Support Circuit)
V_{CC} = +5V ±5%	I^4_{CC} = 2.0A	3.1 A	3.46 A	84 mA + 140 mA/K (2114A-5)
V_{DD} = +12V ±5%	I_{DD} = 150 mA	400 mA	150 mA	Not Required
V_{BB} = −5V ±5%	I_{BB} = 2 mA	200 mA	2 mA	Not Required
V_{AA} = −12V ±5%	I_{AA} = 175 mA	175 mA	175 mA	Not Required

NOTES:

1. Does not include power required for optional ROM/EPROM, I/O drivers, or I/O terminators.
2. With four Intel 2708 EPROMS and 220Ω/330Ω for terminators, installed for 48 input lines. All terminator inputs low.
3. Same as #2 except with four 2758s, 2716s, or 2732s installed.
4. I_{CC} shown without RAM supply current. For 2114A-5 add 140 mA per K byte to a maximum of 560 mA.

Line Drivers and Terminators

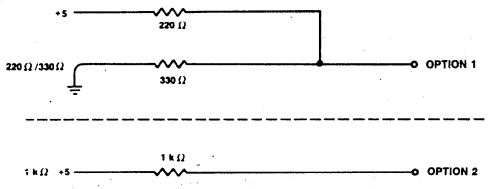
I/O Drivers — The following line drivers and terminators are all compatible with the I/O driver sockets on the iSBC 80/10B Board:

Driver	Characteristic	Sink Current (mA)
7438	I,OC	48
7437	I	48
7432	NI	16
7426	I,OC	16
7409	NI,OC	16
7408	NI	16
7403	I,OC	16
7400	I	16

Note
I - inverting, NI - non-inverting, OC - open collector.

Port 1 has 25 nA totem pole drivers and 1 kΩ terminators.

I/O Terminators — 220Ω/330Ω divider or 1 kΩ pull up.



MULTIBUS® Drivers

Function	Characteristic	Sink Current (mA)
Data	Tri-State	25
Address	Tri-State	25
Commands	Tri-State	25

Environmental Characteristics

Operating Temperature — 0°C to 55°C

Equipment Supplied

iSBC 80/10B Single Board Computer
iSBC 80/10B Schematics

Reference Manual

9803119-01 — iSBC 80/10B Single Board Computer Hardware Reference Manual (NOT SUPPLIED).

Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

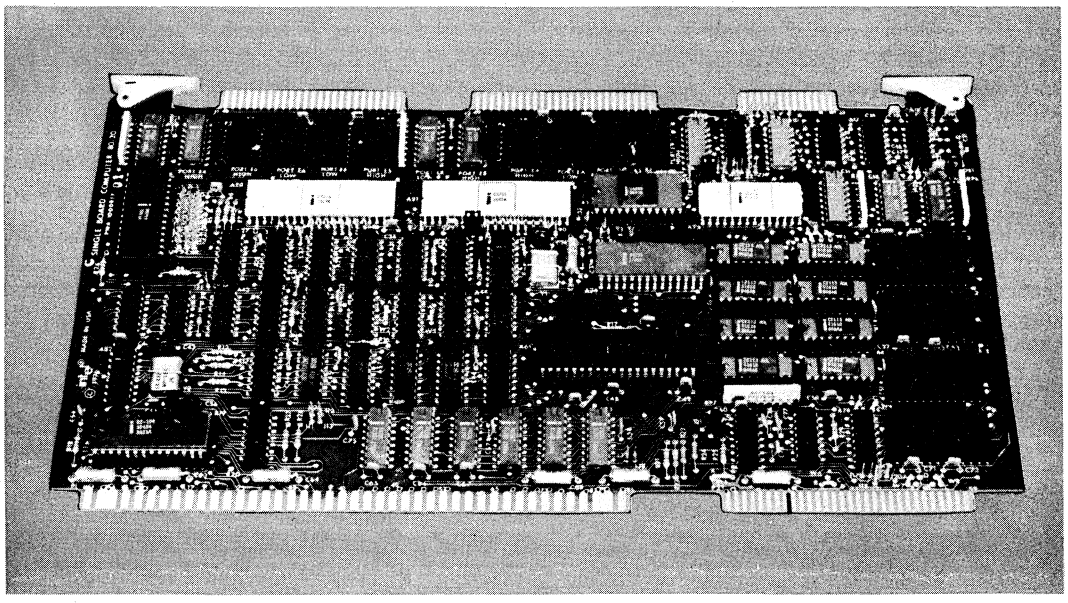
ORDERING INFORMATION

Part Number	Description
SBC 80/10B	Single Board Computer

iSBC® 80/20-4 (or pSBC 80/20-4*) SINGLE BOARD COMPUTER

- 8080A CPU used as central processor
- 4K bytes of static read/write memory
- Sockets for up to 8K bytes of erasable reprogrammable or masked read only memory
- 48 programmable parallel I/O lines with sockets for interchangeable line drivers and terminators
- Programmable synchronous/asynchronous RS232C compatible serial interface with fully software selectable baud rate generation
- Full MULTIBUS® control logic allowing up to 16 masters to share system bus
- Two programmable 16-bit BCD and binary timers
- Eight-level programmable interrupt control
- Compatible with optional memory and I/O expansion boards
- Auxiliary power bus, memory protect, and power-fail interrupt control logic provided for battery backup RAM requirements

The iSBC 80/20-4 Single Board Computer is a member of Intel's complete line of OEM computer systems which take full advantage of Intel's LSI technology to provide economical, self-contained computer-based solutions for OEM applications. Each iSBC 80/20-4 is a complete computer system on a single 6.75 × 12.00-inch printed circuit card. The CPU, system clock, read/write memory, nonvolatile read only memory, I/O ports and drivers, serial communications interface, priority interrupt logic, two programmable timers, MULTIBUS control logic, and bus expansion drivers all reside on each board.



*Same product, manufactured by Intel Puerto Rico, Inc.

October, 1984
AFN-01499B

FUNCTIONAL DESCRIPTION

Intel's powerful 8-bit n-channel MOS 8080A CPU, fabricated on a single LSI chip, is the central processor for the iSBC 80/20-4. The 8080A contains six 8-bit general purpose registers and an accumulator. The six general purpose registers may be addressed individually or in pairs, providing both single and double precision operators. Minimum instruction execution time is 1.86 microseconds. A block diagram of iSBC 80/20-4 functional components is shown in Figure 1.

Memory Addressing

The 8080A has a 16-bit program counter which allows direct addressing of up to 65,536 bytes of memory. An external stack, located within any portion of read/write memory, may be used as a last-in/first-out storage area for the contents of the program counter, flags, accumulator, and all of the six general purpose registers. A 16-bit stack pointer controls the addressing of this external stack. This stack provides subroutine nesting bounded only by memory size.

Memory Capacity

The iSBC 80/20-4 contains 4K bytes of static read/write memory using Intel low power static RAMs. All on-board RAM read and write operations are performed at maximum processor speed. Power for on-board RAM memory is provided on an auxiliary power bus, and memory protect logic is included for battery backup RAM requirements. Sockets for up to 8K bytes of nonvolatile read only memory are provided on the board. Read only memory may be added in 1K-byte increments using Intel 2708 erasable and electrically reprogrammable ROMs (EPROMs), or read only memory may be added in 2K-byte increments using Intel 2716 EPROMs. All on-board ROM read operations are performed at maximum processor speed.

Parallel I/O Interface

The iSBC 80/20-4 contains 48 programmable parallel I/O lines implemented using two Intel 8255 programmable peripheral interfaces. The system software is used to configure the I/O lines in any combination of the unidirectional input/output, and bidirectional ports indicated in Table 1. Therefore, the I/O interface may be customized to meet specified peripheral requirements. In order to take full advantage of the large number of possible I/O configurations, sockets are provided for interchangeable I/O line drivers and terminators. Hence, the flexibility of the I/O interface is further enhanced by the capability of selecting the appropriate combination of optional line drivers and terminators to provide the required sink current, polarity, and drive/termination characteristics for each application. The 48 programmable I/O lines and signal ground lines are brought out to two 50-pin edge connectors that mate with flat, woven, or round cable.

Serial I/O Interface

A programmable communications interface using Intel's 8251 Universal Synchronous/Asynchronous Receiver/Transmitter (USART) is contained on the iSBC

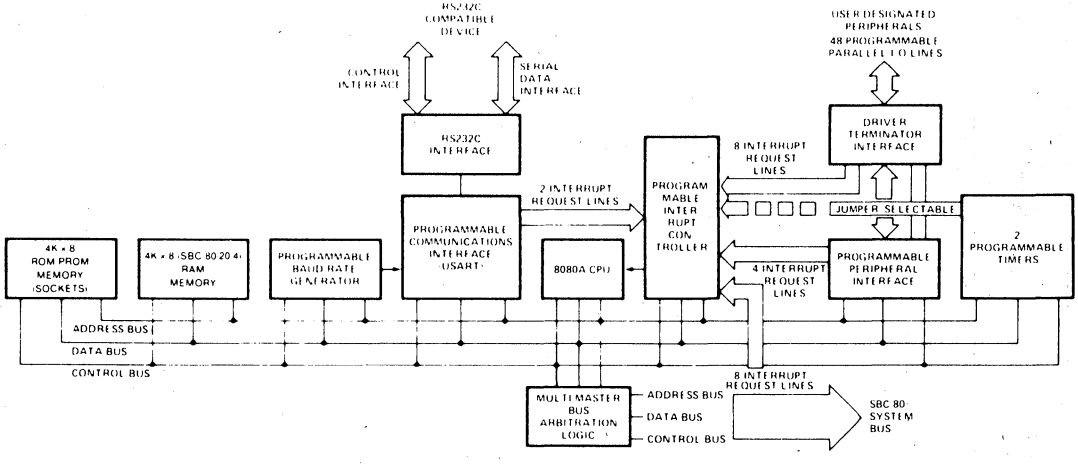


Figure 1. iSBC® 80/20 and iSBC® 80/20-4 Block Diagram Showing Functional Components

80/20-4 board. A software selectable baud rate generator provides the USART with all common communications frequencies. The USART can be programmed by the system software to select the desired asynchronous or synchronous serial data transmission technique (including IBM Bi-Sync). The mode of operation (i.e., synchronous or asynchronous), data format, control character parity, and baud rate are all under program control. The 8251 provides full duplex, double-buffered transmit and receive capability. Parity, overrun, and framing error detection are all incorporated in the USART. The RS232C compatible interface on each board, in conjunction with the USART, provides a direct interface to RS232C compatible terminals, cassettes, and asynchronous and synchronous modems. The RS232C command lines, serial data lines, and signal ground line are brought out to a 26-pin edge connector that mates with RS232C compatible flat or round cable.

Multimaster Capability

The iSBC 80/20-4 is a full computer on a single board with resources capable of supporting the majority of OEM system requirements. For those applications requiring additional processing capacity and the benefits of multiprocessing (i.e., several CPUs and/or controllers logically share system tasks with communication over the system bus), the iSBC 80/20-4 provides full MULTIBUS arbitration control logic. This control logic allows up to three iSBC 80/20-4 or high speed controllers to share the system bus in serial (daisy chain) priority fashion, and up to 16 masters may share the system bus with the addition of an external priority network. Once bus control is attained, a bus bandwidth of up to 5M bytes/sec may be achieved.

The bus controller provides its own clock which is derived independently from the processor clock. This allows different speed controllers to share resources on the same bus, and transfers via the bus proceed asynchronously. Thus, transfer speed is dependent on transmitting and receiving devices only. This design prevents slow master modules from being handicapped in their attempts to gain control of the bus, but does not restrict the speed at which faster modules can transfer data via the same bus. Once a bus request is granted, single or multiple read/write transfers can proceed at a maximum rate of 5 million data words per second. The most obvious applications for the master-slave capabilities of the bus are multiprccessor configurations, high speed direct-memory-access (DMA) operations and high speed peripheral control, but are by no means limited to these three.

Programmable Timers

The iSBC 80/20-4 board provides three fully programmable and independent BCD and binary 16-bit interval timers/event counters utilizing an Intel 8253 Programmable Interval Timer. Two of these timers/counters are available to the systems designer to generate accurate time intervals under software control. Routing of these counters is jumper selectable. Each may be independently routed to the programmable interrupt controller, the I/O line drivers and terminators, or outputs from the 8255 programmable peripheral interfaces. The third interval timer in the 8253 provides the programmable baud

Table 1. Input/Output Port Modes of Operation

Port	Lines (qty)	Mode of Operation					
		Unidirectional				Bidirectional	Control
		Input		Output			
		Unlatched	Latched & Strobed	Latched	Latched & Strobed		
1	8	X	X	X	X	X	
2	8	X	X	X	X		
3	4	X		X			X[1]
	4	X		X			X[1]
4	8	X	X	X	X	X	
5	8	X	X	X	X		
6	4	X		X			X[2]
	4	X		X			X[2]

Notes

1. Part of port 3 must be used as a control port when either port 1 or port 2 are used as a latched and strobed input or a latched and strobed output port or port 1 is used as a bidirectional port.

2. Part of port 6 must be used as a control port when either port 4 or port 5 are used as a latched and strobed input or a latched and strobed output port or port 4 is used as a bidirectional port.

rate generator for the iSBC 80/20-4 RS232C USART serial port. In utilizing the iSBC 80/20-4, the systems designer simply configures, via software, each timer independently to meet system requirements. Whenever a given time delay or count is needed, software commands to the programmable timers/event counters select the desired function. Seven functions are available, as shown in Table 2. The contents of each counter may be read at any time during system operation with simple read operations for event counting applications, and special commands are included so that the contents of each counter can be used "on the fly"

Table 2. Programmable Timer Functions

Function	Operation
Interrupt on terminal count	When terminal count is reached, an interrupt request is generated. This function is extremely useful for generation of real-time clocks.
Programmable one-shot	Output goes low upon receipt of an external trigger edge or software command and returns high when terminal count is reached. This function is retriggerable.
Rate generator	Divide by N counter. The output will go low for one input clock cycle, and the period from one low-going pulse to the next is N times the input clock period.
Square-wave rate generator	Output will remain high until one-half the count has been completed, and go low for the other half of the count.
Software triggered strobe	Output remains high until software loads count (N). N counts after count is loaded, output goes low for one input clock period.
Hardware triggered strobe	Output goes low for one clock period N counts after rising edge on counter trigger input. The counter is retriggerable.
Event counter	On a jumper selectable basis, the clock input becomes an input from the external system. CPU may read the number of events occurring after the counting "window" has been enabled or an interrupt may be generated after N events occur in the system.

Interrupt Capability

Operation and Priority Assignments — An Intel 8259 Programmable Interrupt Controller (PIC) provides vectoring for eight interrupt levels. As shown in Table 3, a selection of four priority processing modes is available to the systems designer so that the manner in which requests are processed may be configured to match system requirements. Operating mode and priority assignments may be reconfigured dynamically via software at any time during system operation. The PIC accepts interrupt requests from the programmable parallel and serial I/O interfaces, the programmable timers, the system bus, or directly from peripheral equipment. The PIC then determines which of the incoming requests is of the highest priority, determines whether this request is of higher priority than the level currently being serviced, and if appropriate, issues an interrupt to the CPU. Any combination of interrupt levels may be masked through storage via software, of a single byte to the interrupt register of the PIC.

Table 3. Programmable Interrupt Modes

Mode	Operation
Fully nested	Interrupt request line priorities fixed at 0 as highest, 7 as lowest.
Auto-rotating	Equal priority. Each level, after receiving service, becomes the lowest priority level until the next interrupt occurs.
Specific priority	System software assigns lowest priority level. Priority of all other levels based in sequence numerically on this assignment.
Polled	System software examines priority-encoded system interrupt status via interrupt status register.

Interrupt Addressing — The PIC generates a unique memory address for each interrupt level. These addresses are equally spaced at intervals of 4 or 8 (software selectable) bytes. This 32- or 64-byte block may be located to begin at any 32- or 64-byte boundary in the 65,536-byte memory space. A single 8080 jump instruction at each of these addressed then provides linkage to locate each interrupt service routine independently anywhere in memory.

Interrupt Request Generation — Interrupt requests may originate from 26 sources. Four jumper selectable interrupt requets can be automatically generated by the programmable peripheral interface when a byte of information is ready to be transferred to the CPU (i.e., input buffer is full) or a byte of information has been transferred to a peripheral device (i.e., output buffer is empty). Two jumper selectable interrupt requests can be automatically generated by the USART when a character is ready to be transfer to the CPU (i.e., receive channel buffer is full), or a character is ready to be transmitted (i.e., transmit channel data buffer is empty). A jumper selectable request can be generated by each of the programmable timers. Nine additional interrupt request lines are available to the user for direct interface to user designated peripheral devices via the system bus, and eight interrupt request lines may be jumper routed directly from peripherals via the parallel I/O driver/terminator section.

Power-Fail Control — Control logic is also included for generation of a power-fail interrupt which works in conjunction with the AC-low signal from iSBC 635 Power Supply or equivalent.

Expansion Capabilities

Memory and I/O capacity may be expanded and additional functions added using Intel MULTIBUS compatible expansion boards. High speed integer and floating-point arithmetic capabilities may be added by using the iSBC 310A High Speed Mathematics Unit. Memory may be expanded to 65,536 bytes by adding user specified combinations of RAM boards, EPROM boards, or combination boards. Input/output capacity may be increased by adding digital I/O and analog I/O expansion boards. Mass storage capability may be achieved by adding single or double density diskette controllers as subsystems. Modular expandable backplanes and cardcages are available to support multiboard systems.

System Development Capability

The development cycle of iSBC 80/20-4-based products may be significantly reduced using Intel's system development tools available today. For those not requiring hardware emulation capability, Intel provides a new low cost microcomputer development system. The iPDS, Personal Development System, provides low cost system development for the iSBC 80/20-4 board, while at the same time providing personal computer capability for the engineer. The Intellec Series II family of compatible microcomputer development systems provides a range of capability from a low cost disk-based edit debug workstation to a high performance, fully compatible hard-disk-based software development system. A unique in-circuit emulator (ICE-80) option provides the capability of developing and debugging software directly on the iSBC 80/20-4 board.

Programming Capability

PL/M-80 — Intel's high level programming language, PL/M, is also available as a resident Intellec microcomputer development system option. PL/M provides the capability to program in a natural, algorithmic language and eliminates the need to manage register usage or allocate memory. PL/M programs can be written in a much shorter time than assembly langauge programs for a given application.

FORTRAN-80 — For applications requiring computational and formatted I/O capabilities, the high level FORTRAN-80 programming language is also available as a resident option of the Intellec system. The FORTRAN compiler produces relocatable object code that may be easily linked with PL/M or assembly language program modules. This gives the user a wide flexibility in developing software.

BASIC-80 — A high level language interpreter with extended disk capabilities which operates under the RMX/80 Real-Time Multi-Tasking Executive and translates BASIC-80 source programs into an internally executable form. This language interpreter, provided as a set of linkable object modules, is ideally suited to the OEM who requires a pass through programming language. The BASIC-80 programs may be created, stored and interpreted on the iSBC 80 based system. The BASIC-80 language has a rich complement of statements, functions, and commands to program applications requiring a full range of 1) string manipulation and disk I/O for data processing, 2) single and double precision floating point and array handling for numeric analysis, or 3) port I/O with mask operations controlled through bit-wise Boolean logical operators.

SPECIFICATIONS

Word Size

Instruction — 8, 16, or 24 bits
Data — 8 bits

Cycle Time

Basic Instruction Cycle — 1.86 μs

Note
Basic instruction cycle is defined as the fastest instruction (i.e., four clock cycles).

Memory Addressing

On-Board ROM/EPROM — 0-0FFF (2708) or 0-1FFF (2716)

On-Board RAM — 4K bytes ending on a 16K boundary (e.g., $3FFF_H$, $7FFF_H$, $BFFF_H$, . . . $FFFF_H$)

Memory Capacity

On-Board ROM/EPROM — 8K bytes (sockets only)

On-Board RAM — 4K bytes

Off-Board Expansion — Up to 65,536 bytes in user specified RAM, ROM, and EPROM

Note
ROM/EPROM may be added in 1K or 2K-byte increments.

I/O Addressing

On-Board Programmable I/O (see Table 1)

Port	8255 No. 1			8255 No. 2			8255 No. 1 Control	8255 No. 2 Control	USART Data	USART Control
	1	2	3	4	5	6				
Address	E4	E5	E6	E8	E9	EA	E7	EB	EC	ED

I/O Capacity

Parallel — 48 programmable lines (see Table 1)

Note
Expansion to 504 input and 504 output lines can be accomplished using optional I/O boards.

Serial Communications Characteristics

Synchronous — 5-8 bit characters; internal or external character synchronization; automatic sync insertion

Asynchronous — 5-8 bit characters; break character generation; 1, 1½, or 2 stop bits; false start bit detection

Baud Rates

Frequency (kHz) (Software Selectable)	Baud Rate (Hz)		
	Synchronous	Asynchronous	
		÷ 16	÷ 64
153.6	—	9600	2400
76.8	—	4800	1200
38.4	38400	2400	600
19.2	19200	1200	300
9.6	9600	600	150
4.8	4800	300	75
2.4	2400	150	—
1.76	1760	110	—

Note
Frequency selected by I/O write of appropriate 16-bit frequency factor to baud rate register.

Register Address (hex notation, I/O address space)

DE Baud rate register

Note
Baud rate factor (16 bits) is loaded as two sequential output operations to same address (DE_H).

Interrupts

Register Addresses (hex notation, I/O address space)

DA Interrupt request register
DA In-service register
DB Mask register
DA Command register
DB Block address register
DA Status (polling register)

Note
Several registers have the same physical address; sequence of access and one data bit of control word determine which register will respond.

Timers

Register Addresses (hex notation, I/O address space)

DF Control register
DC Timer 1
DD Timer 2

Note
Timer counts loaded as two sequential output operations to same address, as given.

Input Frequencies

Reference	Event Rate
1.0752 MHz ± 10% (0.930 μs period, nominal)	1.1 MHz max

Note
Maximum rate for external events in event counter function.

Output Frequencies/Timing Intervals

Function	Single Timer/Counter		Dual Timer/Counter (Two Timers Cascaded)	
	Min	Max	Min	Max
Real-time interrupt	1.86 μs	60.948 ms	3.72 μs	1.109 hr
Programmable one-shot	1.86 μs	60.948 ms	3.72 μs	1.109 hr
Rate generator	16.407 Hz	537.61 kHz	0.00025 Hz	268.81 kHz
Square-wave rate generator	16.407 Hz	537.61 kHz	0.00025 Hz	268.31 kHz
Software triggered strobe	1.86 μs	60.948 ms	3.72 μs	1.109 hr
Hardware triggered strobe	1.86 μs	60.948 ms	3.72 μs	1.109 hr

Interfaces

Bus — All signals TTL compatible
Parallel I/O — All signals TTL compatible
Interrupt Requests — All TTL compatible
Timer — All signals TTL compatible
Serial I/O — RS232C compatible, data set configuration

System Clock (8080A CPU)

2.1504 MHz ± 0.1%

Auxiliary Power

An auxiliary power bus is provided to allow separate power to RAM for systems requiring battery backup of read/write memory. Selection of this auxiliary RAM power bus is made via jumpers on the board.

Memory Protect

An active-low TTL compatible memory protect signal is brought out on the auxiliary connector which, when asserted, disables read/write access to RAM memory on the board. This input is provided for the protection of RAM contents during system power-down sequences.

Connectors

Interface	Double-Sided Pins (qty)	Centers (in.)	Mating Connectors*
MULTIBUS System Bus	86	0.156	ELFAB BS1562043PBB Viking 2KH43/9AMK12 Soldered PCB Mount EDAC 337086540201 ELFAB BW1562D43PBB EDAC 337086540202 ELFAB BW1562A43PBB Wire Wrap
Auxiliary Bus	60	0.100	EDAC 345060524802 ELFAB BS1020A30PBB EDAC 345060540201 ELFAB BW1020D30PBB Wire Wrap
Parallel I/O (2)	50	0.100	3M 3415-001 Flat Crimp GTE Sylvania 6AD01251A1DD Soldered
Serial I/O	26	0.100	AMP 15837151 EDAC 345026520202 PCB Soldered 3M 3462-0001 AMP 88373-5 Flat Crimp

***Note:** Connectors compatible with those listed may also be used.

Line Drivers and Terminators

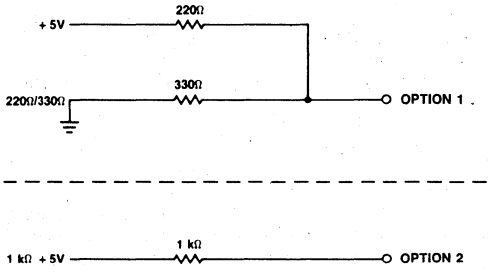
I/O Drivers — The following line drivers are all compatible with the I/O driver sockets on the iSBC 80/20-4.

Driver	Characteristic	Sink Current (mA)
7438	I,OC	48
7437	I	48
7432	NI	16
7426	I,OC	16
7409	NI,OC	16
7408	NI	16
7403	I,OC	16
7400	I	16

Note
I = inverting; NI = non-inverting; OC = open collector.

Ports 1 and 4 have 20 mA totem-pole bidirectional drivers and 1 kΩ terminators.

I/O Terminators — 220Ω/330Ω divider or 1 kΩ pull-up



Bus Drivers

Driver	Characteristic	Sink Current (mA)
Data	Tri-state	50
Address	Tri-state	50
Commands	Tri-state	32

Physical Characteristics

Width — 12.00 in. (30.48 cm)
Height — 6.75 in. (17.15 cm)
Depth — 0.50 in. (1.26 cm)
Weight — 14 oz (397.6 gm)

Electrical Characteristics

DC Power Requirements

Voltage (±5%)	Without PROM[1] (max)	With 4K PROM[2] (max)	With iSBC 530[3] (max)	RAM Only[4] (max)	With 8K PROM[5] (max)
V_{CC} = +5V	I_{CC} = 4.0A	4.9A	4.9A	1.1A	5.2A
V_{DD} = +12V	I_{DD} = 90 mA	350 mA	450 mA	—	90 mA
V_{BB} = −5V	I_{BB} = 2 mA	180 mA	180 mA	—	2 mA
V_{AA} = −12V	I_{AA} = 20 mA	20 mA	120 mA	—	20 mA

Notes

1. Does not include power required for optional PROM, I/O drivers, and I/O terminators.
2. With four 2708 EPROMs and 220Ω/330Ω input terminators installed for 32 I/O lines, all terminator inputs low.
3. With four 2708 EPROMs, 220Ω/330Ω input terminators installed for 32 I/O lines, all terminator inputs low, and iSBC 530 Teletypewriter Adapter drawing power from serial port connector.
4. RAM chips powered via auxiliary power bus.
5. With four 8716 EPROMs and eight 220Ω/330Ω input terminators installed, all terminator inputs low.

Environmental Characteristics

Operating Temperature — 0°C to 55°CC

Reference Manual

9800317D — iSBC 80/20-5 Hardware Reference Manual (NOT SUPPLIED)

Reference manuals are shipped with each product only if designated SUPPLIED (see above). Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

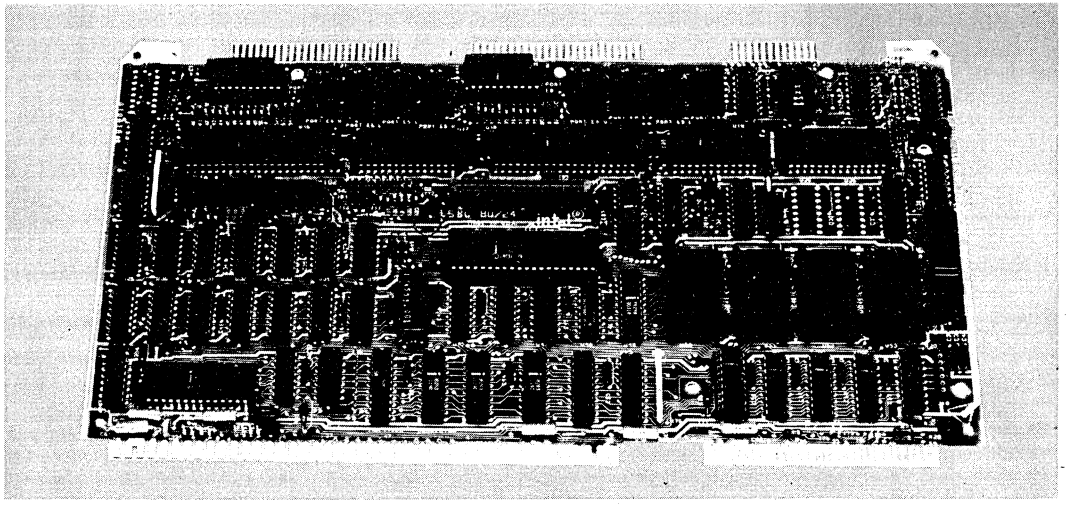
ORDERING INFORMATION

Part Number	Description
SBC 80/20-4	Single Board Computer with 4K bytes RAM

iSBC® 80/24 (or pSBC 80/24*) SINGLE BOARD COMPUTER

- **Upward compatible with iSBC 80/20-4 Single Board Computer**
- **8085A-2 CPU operating at 4.8 or 2.4 MHz**
- **Two iSBX™ bus connectors for iSBX ™ MULTIMODULE™ board expansion**
- **4K bytes of static read/write memory expandable on-board to 8K bytes using the iSBC 301 MULTIMODULE ™ Board**
- **Sockets for up to 32K bytes of read only memory**
- **48 programmable parallel I/O lines with sockets for interchangeable line drivers and terminators**
- **Programmable synchronous/asynchronous RS232C compatible serial interface with software selectable baud rates**
- **Full MULTIBUS® control logic for multimaster configurations and system expansion**
- **Two programmable 16-bit BCD or binary timers/event counters**
- **12 levels of programmable interrupt control**
- **Auxiliary power bus, memory protect, and power-fail interrupt control logic provided for battery backup RAM requirements**

The Intel® iSBC 80/24 Single Board Computer is a member of Intel's complete line of OEM microcomputer systems which take full advantage of Intel's LSI technology to provide economical, self-contained computer-based solutions for OEM applications. The iSBC 80/24 board is a complete computer system on a single 6.75 x 12.00-inch printed circuit card. The CPU, system clock, iSBX bus interface, read/write memory, read only memory sockets, I/O ports and drivers, serial communications interface, priority interrupt logic, and programmable timers all reside on the board. Full MULTIBUS interface logic is included to offer compatibility with the Intel OEM Microcomputer Systems family of Single Board Computers, expansion memory options, digital and analog I/O expansion boards, and peripheral and communications controllers.



*Same product, manufactured by Intel Puerto Rico, Inc.

October, 1984
Order Number: 142927-002

FUNCTIONAL DESCRIPTION

Central Processing Unit

Intel's powerful 8-bit N-channel 8085A-2 CPU fabricated on a single LSI chip, is the central processor for the iSBC 80/24 board operating at either 4.8 or 2.4 MHz (jumper selectable). The 8085A-2 CPU is directly software compatible with the Intel 8080A CPU. The 8085A-2 contains six 8-bit general purpose registers and an accumulator. The six general purpose registers may be addressed individually or in pairs, providing single and double precision operators. Minimum instruction execution time is 826 nanoseconds. A block diagram of the iSBC 80/24 functional components is shown in Figure 1.

MULTIMODULE™ Board Expansion

The iSBX bus interface brings designers incremental on-board expansion at minimal cost. Two iSBX bus MULTIMODULE connectors are provided for plug-in expansion of any iSBX MULTIMODULE board. The iSBX MULTIMODULE concept provides the ability to adapt quickly to new technology, the economy of buying only what is needed, and the ready availability of a spectrum of functions for greater application potential. iSBX boards are available to provide expansion equivalent to the I/O available on the iSBC 80/24 board or the user may configure entirely new functionality, such as math, directly on board. The iSBX 350 Parallel I/O MULTIMODULE board provides 24 I/O lines using an 8255A Programmable Peripheral Interface. Therefore, two iSBX 350 modules together with the iSBC 80/24 board may offer 96 lines of programmable I/O. Alternately, a serial port may be added using the iSBX 351 Serial I/O MULTIMODULE board and math may be configured on-board with the iSBX 332 Floating Point Math or iSBX 331 Fixed/Floating Point Math MULTIMODULE board. Future iSBX products are also planned. The iSBX MULTIMODULE board is a logical extension of the on-board programmable I/O and is accessed by the iSBC 80/24 single board computer as common I/O port locations. The iSBX board is coupled directly to the 8085A-2 CPU and therefore becomes an integral element of the iSBC 80/24 single board computer providing optimum performance. In addition, RAM memory capacity may be expanded to 8K bytes using the iSBC 301 4K Byte RAM MULTIMODULE board. All MULTIMODULE boards ranging from the iSBC 301 module to the iSBX modules offer incremental expansion, optimum performance, and minimal cost.

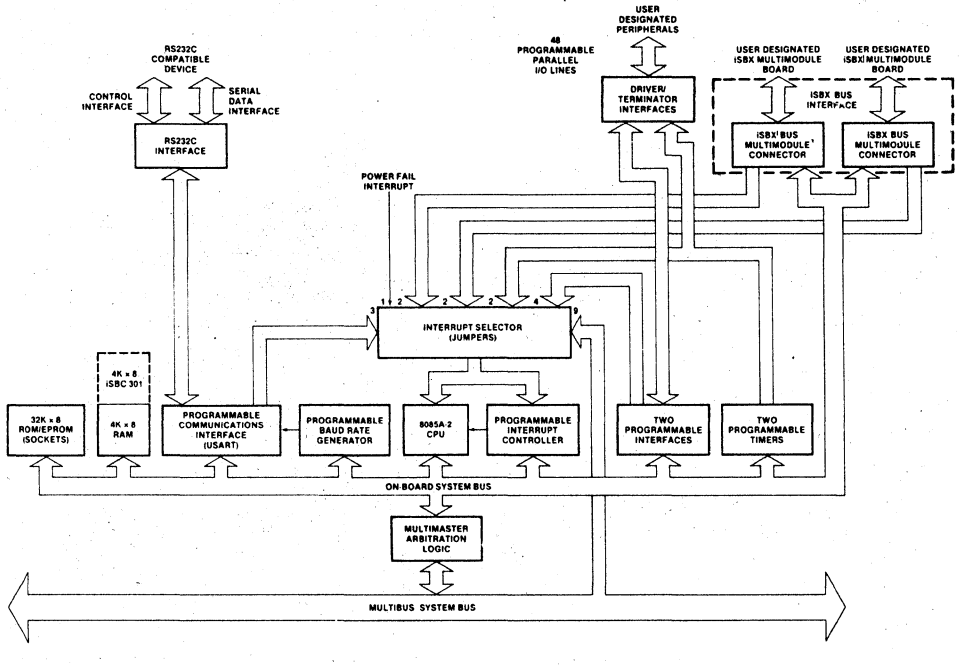


Figure 1. iSBC® 80/24 Single Board Computer Block Diagram

Memory Addressing

The 8085A-2 has a 16-bit program counter which allows direct addressing of up to 64K bytes of memory. An external stack, located within any portion of read/write memory, may be used as a last-in/first-out storage area for the contents of the program counter, flags, accumulator, and all of the six general purpose registers. A 16-bit stack pointer controls the addressing of this external stack. This stack provides subroutine nesting bounded only by memory size.

Memory Capacity

The iSBC 80/24 board contains 4K bytes of static read/write memory using Intel 8185-2 RAMs. In addition, the on-board RAM capacity may be expanded to 8K bytes with the iSBC 301 4K byte RAM MULTIMODULE board. All RAM read and write operations are performed at maximum processor speed. Power for the on-board RAM may be provided on an auxiliary power bus, and memory protect logic is included for RAM battery backup requirements.

Four sockets are provided for up to 32K bytes of nonvolatile read only memory on the iSBC 80/24 board. EPROM may be added in 1K byte increments up to 4K bytes (using Intel 2708 or 2758); in 2K byte increments up to 8K bytes (using Intel 2716); in 4K byte increments up to 16K bytes (using Intel 2732); or in 8K byte increments up to 32K bytes (using Intel 2764).

Parallel I/O Interface

The iSBC 80/24 board contains 48 programmable parallel I/O lines implemented using two Intel 8255A Programmable Peripheral Interfaces. The system software is used to configure the I/O lines in any combination of unidirectional input/output and bidirectional ports as indicated in Table 1. Therefore, the I/O interface may be customized to meet specific peripheral requirements. In order to take full advantage of the large number of possible I/O configurations, sockets are provided for interchangeable I/O line drivers and terminators. Hence, the flexibility of the I/O interface is further enhanced by the capability of selecting the appropriate combination of optional line drivers and terminators to provide the required sink current, polarity, and drive/termination characteristics for each application. The 48 programmable I/O lines and signal ground lines are brought out to two 50-pin edge connectors that mate with flat, woven, or round cables.

Serial I/O Interface

A programmable communications interface using the Intel 8251A Universal Synchronous/Asynchronous Receiver/Transmitter (USART) is contained on the iSBC 80/24 board. A software selectable baud rate generator provides the USART with all common communication frequencies. The USART can be programmed by the system software to select the desired asynchronous or synchronous serial data transmission technique (including IBM

Table 1. Input/Output Port Modes of Operation

Port	Lines (qty)	Mode of Operation					Control
		Unidirectional				Bidirectional	
		Input		Output			
		Unlatched	Latched & Strobed	Latched	Latched & Strobed		
1	8	X	X	X	X	X	
2	8	X	X	X	X		
3	4	X		X			X[1]
	4	X		X			X[1]
4	8	X	X	X	X	X	
5	8	X	X	X	X		
6	4	X		X			X[2]
	4	X		X			X[2]

NOTES:

1. Part of port 3 must be used as a control port when either port 1 or port 2 are used as a latched and strobed input or a latched and strobed output port or port 1 is used as a bidirectional port.
2. Part of port 6 must be used as a control port when either port 4 or port 5 are used as a latched and strobed input or a latched and strobed output port or port 4 is used as a bidirectional port.

Bi-Sync). The mode of operation (i.e. synchronous or asynchronous), data format, control character format, parity, and baud rate are all under program control. The 8251A provides full duplex, double buffered transmit and receive capability. Parity, overrun, and framing error detection are all incorporated in the USART. The RS232C compatible interface, in conjunction with the USART, provides a direct interface to RS232C compatible terminals, cassettes, and asynchronous and synchronous modems. The RS232C command lines, serial data lines, and signal ground line are brought out to a 26-pin edge connector that mates with RS232C compatible flat or round cable.

Multimaster Capability

The iSBC 80/24 board is a full computer on a single board with resources capable of supporting a large variety of OEM system requirements. For those applications requiring additional processing capacity and the benefits of multiprocessing (i.e. several CPUs and/or controllers logically sharing system tasks through communication over the system bus), the iSBC 80/24 board provides full MULTIBUS arbitration control logic. This control logic allows up to three iSBC 80/24 boards or other bus masters to share the system bus in serial (daisy chain) priority fashion, and up to 16 masters to share the MULTIBUS system bus with the addition of an external priority network. The MULTIBUS arbitration logic operates synchronously with a MULTIBUS clock (provided by the iSBC 80/24 board or optionally connected directly to the MULTIBUS clock) while data is transferred via a handshake between the master and slave modules. This allows different speed controllers to share resources on the same bus since transfers via the bus proceed asynchronously. Thus, transfer speed is dependent on transmitting and receiving devices only. This design provides slow master modules from being handicapped in their attempts to gain control of the bus, but does not restrict the speed at which faster modules can transfer data via the same bus. The most obvious applications for the master-slave capabilities of the bus are multiprocessor configurations, high speed direct memory access (DMA) operations, and high speed peripheral control, but are by no means limited to these three.

Programmable Timers

The iSBC 80/24 board provides three independent, fully programmable 16-bit interval timers/event counters utilizing the Intel 8253 Programmable Interval Timer. Each counter is capable of operating in either BCD or binary modes. Two of these timers/counters are available to the systems designer to generate accurate time intervals under software control. Routing for the outputs and gate/trigger inputs of two of these counters is jumper selectable. The outputs may be independently routed to the 8259A Programmable Interrupt Controller, to the I/O line drivers associated with the 8255A Programmable Peripheral Interface, or may be routed as inputs to the 8255A chip. The gate/trigger inputs may be routed to I/O terminators associated with the 8255A or as output connections from the 8255A. The third interval timer in the 8253 provides the programmable baud rate generator for the RS232C USART serial port. In utilizing the iSBC 80/24 board, the systems designer simply configures, via software, each timer independently to meet system require-

Table 2. Programmable Timer Functions

Function	Operation
Interrupt on terminal count	When terminal count is reached, an interrupt request is generated. This function is extremely useful for generation of real-time clocks.
Programmable one-shot	Output goes low upon receipt of an external trigger edge or software command and returns high when terminal count is reached. This function is retriggerable.
Rate generator	Divide by N counter. The output will go low for one input clock cycle, and the period from one low-going pulse to the next is N times the input clock period.
Square-wave rate generator	Output will remain high until one-half the count has been completed, and go low for the other half of the count.
Software triggered strobe	Output remains high until software loads count (N). N counts after count is loaded, output goes low for one input clock period.
Hardware triggered strobe	Output goes low for one clock period N counts after rising edge on counter trigger input. The counter is retriggerable.
Event counter	On a jumper selectable basis, the clock input becomes an input from the external system. CPU may read the number of events occurring after the counting "window" has been enabled or an interrupt may be generated after N events occur in the system.

ments. Whenever a given time delay or count is needed, software commands to the programmable timers/event counters select the desired function. Seven functions are available, as shown in Table 2. The contents of each counter may be read at any time during system operation with simple read operations for event counting applications, and special commands are included so that the contents of each counter can be read "on the fly".

Interrupt Capability

The iSBC 80/24 board provides vectoring for 12 interrupt levels. Four of these levels are handled directly by the interrupt processing capability of the 8085A-2 CPU and represent the four highest priority interrupts of the iSBC 80/24 board. Requests are routed to the 8085A-2 interrupt inputs—TRAP, RST 7.5, RST 6.5, and RST 5.5 (in decreasing order of priority), each of which generates a call instruction to a unique address (TRAP: 24H; RST 7.5: 3CH; RST 6.5: 34H; and RST 5.5: 2CH). An 8085A-2 JMP instruction at each of these addresses then provides linkage to interrupt service routines located independently anywhere in memory. All interrupt inputs with the exception of the trap interrupt may be masked via software. The trap interrupt should be used for conditions such as power-down sequences which require immediate attention by the 8085A-2 CPU. The Intel 8259A Programmable Interrupt Controller (PIC) provides vectoring for the next eight interrupt levels. As shown in Table 3, a selection of four priority processing modes is available to the systems designer for use in designing request processing configurations to match system requirements. Operating mode and priority assignments may be reconfigured dynamically via software at any time during system operation. The PIC accepts interrupt requests from the programmable parallel and serial I/O interfaces, the programmable timers, the system bus, iSBX bus, or directly from peripheral equipment. The PIC then determines which of the incoming requests is of the highest priority, determines whether this request is of higher priority than the level currently being serviced, and, if appropriate, issues an interrupt to the CPU. Any combination of interrupt levels may be masked, via software, by storing a single byte in the interrupt mask register of the PIC. The PIC generates a unique memory address for each interrupt level. These addresses are equally spaced at intervals of 4 or 8 (software selectable) bytes. This 32 or 64-byte block may be located to begin at any 32 or 64-byte boundary in the 65,536-byte memory space. A single 8085A-2 JMP instruction at each of these addresses then provides linkage to locate each interrupt service routine independently anywhere in memory.

Table 3. Programmable Interrupt Modes

Mode	Operation
Fully nested	Interrupt request line priorities fixed at 0 as highest, 7 as lowest.
Autorotating	Equal priority. Each level, after receiving service, becomes the lowest priority level until next interrupt occurs.
Specific priority	System software assigns lowest priority level. Priority of all other levels based in sequence numerically on this assignment.
Polled	System software examines priority-encoded system interrupt status via interrupt status register.

Interrupt Request Generation

Interrupt requests may originate from 23 sources. Two jumper selectable interrupt requests can be generated by each iSBX MULTIMODULE board. Two jumper selectable interrupt requests can be automatically generated by each programmable peripheral interface when a byte of information is ready to be transferred to the CPU (i.e., input buffer is full) or a byte of information has been transferred to a peripheral device (i.e., output buffer is empty). Three jumper selectable interrupt requests can be automatically generated by the USART when a character is ready to be transferred to the CPU (i.e., receiver channel buffer is full), a character is ready to be transmitted (i.e., the USART is ready to accept a character from the CPU), or when the transmitter is empty (i.e., the USART has no character to transmit). A jumper selectable request can be generated by each of the programmable timers. Nine interrupt request lines are available to the user for direct interface to user designated peripheral devices via the MULTIBUS system bus. A power-fail signal can also be selected as an interrupt source.

Power-Fail Control

A power-fail interrupt may be detected through the AC-low signal generated by the power supply. This signal may be configured to interrupt the 8085A-2 CPU to initiate an orderly power down instruction sequence.

MULTIBUS® System Expansion Capabilities

Memory and I/O capacity may be expanded and additional functions added using Intel MULTIBUS system compatible expansion boards. Memory may be expanded to 65,536 bytes by adding user specified combinations of RAM boards, EPROM boards, or combination boards. Input/output capa-

city may be increased by adding digital I/O and analog I/O expansion boards. Mass storage capability may be achieved by adding single or double density diskette or hard disk controllers as subsystems. Expanded communication needs can be handled by communication controllers. Modular expandable backplanes and card cages are available to support multiboard systems.

System Development Capability

The development cycle of iSBC 80/24-based products may be significantly reduced using Intel's system development tools available today. For those not requiring hardware emulation capability, Intel provides a new low cost microcomputer development system. The iPDS, Personal Development System, provides low cost system development for iSBC 80/24 board, while at the same time providing personal computer capability for the engineer. The Intellec Series II family of compatible microcomputer development systems provides a range of capability from a low cost disk-based edit debug workstation to a high performance, fully compatible hard-disk-based software development system. A unique in-circuit emulator (ICE-85A) option provides the capability of developing and debugging software directly on the iSBC 80/24 board.

Programming Capability

PL/M-80—Intel's high level system programming language, PL/M, is also available as a resident Intellec microcomputer development system option. PL/M provides the capability to program in a natural, algorithmic language and eliminates the need to manage register usage or allocate memory. PL/M programs can be written in a much shorter time than assembly language programs.

FORTRAN-80—For applications requiring computational and formatted I/O capabilities, the ANSI 77 standard high level FORTRAN-80 programming language is available as a resident option of the Intellec system. The FORTRAN compiler produces relocatable object code that may be easily linked with PL/M or assembly language program modules. In addition, the iSBC 801 FORTRAN-80 Run-Time Package is a complete, ready-to-use set of linkable object modules which are fully compatible with iRMX 80 systems. The modules, when combined with the FORTRAN-80 coded application, provide the appropriate interfaces to the disk file and terminal I/O of iRMX 80.

BASIC-80—A high level language interpreter is available with extended disk capabilities which operates under the iRMX 80 Real-Time Multitasking Executive and translates BASIC-80 source programs into an internally executable form. This language interpreter, provided as a set of linkable object modules, is ideally suited to the OEM who requires a pass through programming language. The BASIC-80 programs may be created, stored, and interpreted on the iSBC 80-based systems using the iSBC 802 BASIC-80 Configurable iRMX 80 Disk-Based Interpreter. The iSBC 802 Interpreter has a complete ready-to-use set of linkable object modules which are fully compatible with Intel's iRMX 80 Real-Time Multitasking Executive Software. The modules provide interfaces to disk file and terminal I/O, software floating point, or interface to other routines provided by the user.

SPECIFICATIONS

Word Size

Instruction—8, 16, or 24 bits
Data—8 bits

Cycle Time

Basic Instruction Cycle

826 nsec (4.84 MHz operating frequency)
1.65 μsec (2.42 MHz operating frequency)

NOTE:
Basic instruction cycle is defined as the fastest instruction (i.e., four clock cycles).

Memory Addressing

On-Board EPROM

0-0FFF using 2708, 2758 (1 wait state)
0-1FFF using 2716 (1 wait state)
0-3FFF using 2732 (1 wait state)
using 2732A (no wait states)
0-7FFF using 2764A (no wait states)

On-Board RAM

3000-3FFF with no RAM expansion
2000-3FFF with optional RAM (iSBC 301 board)

NOTE:
Default configuration—may be reconfigured to top end of any 16K boundary.

Memory Capacity

On-Board EPROM

32K bytes (sockets only)

May be added in 1K (using Intel 2708 or 2758), 2K (using Intel 2716), 4K (using Intel 2732), or 8K (using Intel 2764) byte increments.

On-Board RAM

4K bytes (8K bytes using iSBC 301 4K byte RAM MULTIMODULE Board)

Off-Board Expansion

Up to 64K bytes using user specified combinations of RAM, ROM, and EPROM.

Up to 128K bytes using bank select control via I/O port and 2 jumper options.

May be disabled using PROM ENABLE via I/O port and jumper option, resulting in off-board RAM overlay capability.

I/O Addressing

On-Board Programmable I/O

Device	I/O Address
8255A No. 1	
Port A	E4
Port B	E5
Port C	E6
Control	E7
8255A No. 2	
Port A	E8
Port B	E9
Port C	EA
Control	EB
8251A	
Data	EC, EE
Control	ED, EF
iSBX MULTIMODULE J5	
MCS0	C0-C7
MCS1	C8-CF
iSBX MULTIMODULE J6	
MCS0	F0-F7
MCS1	F8-FF

I/O Capacity

Parallel—48 programmable lines

Serial—1 transmit, 1 receive, 1 SID, 1 SOD

iSBX MULTIMODULE — 2 iSBX MULTIMODULE Boards

Serial Communications Characteristics

Synchronous—5-8 bit characters; internal or external character synchronization; automatic sync insertion

Asynchronous—5-8 bit characters; break character generation; 1, 1½, or 2 stop bits; false start bit detectors

Baud Rates

Output Frequency in kHz	Baud Rate (Hz)		
	Synchronous	Asynchronous	
		÷16	÷64
153.6	—	9600	2400
76.8	—	4800	1200
38.4	38400	2400	600
19.2	19200	1200	300
9.6	9600	600	150
4.8	4800	300	75
2.4	2400	150	—
1.76	1760	110	—

NOTE:
Frequency selected by I/O write of appropriate 16-bit frequency factor to baud rate register.

Register Address (hex notation, I/O address space)

DE Baud rate register

NOTE:
Baud rate factor (16 bits) is loaded as two sequential output operations to same address (DE_H).

Interrupts

Addresses for 8259A Registers (hex notation, I/O address space)

DA or D8 Interrupt request register
DA or D8 In-service register
DB or D9 Mask register
DA or D8 Command register
DB or D9 Block address register
DA or D8 Status (polling register)

NOTE:
Several registers have the same physical address; sequence of access and one data bit of control word determine which register will respond.

Interrupt levels routed to 8085A-2 CPU automatically vector the processor to unique memory locations:

Interrupt Input	Memory Address	Priority	Type
TRAP	24	Highest	Non-maskable
RST 7.5	3C		Maskable
RST 6.5	34	↓	Maskable
RST 5.5	2C	Lowest	Maskable

Timers

Register Addresses (hex notation, I/O address space)

DF Control register
DC Timer 0
DD Timer 1
DE Timer 2

NOTE:
Timer counts loaded as two sequential output operations to same address as given.

Input Frequencies

Reference: 1.0752 MHz ±0.1% (0.930 μsec period, nominal)

Event Rate: 1.1 MHz max.

Output Frequencies/Timing Intervals

Function	Single Timer/Counter		Dual Timer/Counter (Two Timers Cascaded)	
	Min.	Max.	Min.	Max.
Real-Time Interrupt	1.86 μsec	60.948 msec	3.72 μsec	1.109 hrs
Programmable One-Shot	1.86 μsec	60.948 msec	3.72 μsec	1.109 hrs
Rate Generator	16.407 Hz	537.61 kHz	0.00025 Hz	268.81 kHz
Square-Wave Rate Generator	16.407 Hz	537.61 kHz	0.00025 Hz	268.81 kHz
Software Triggered Strobe	1.86 μsec	60.948 msec	3.72 μsec	1.109 hrs
Hardware Triggered Strobe	1.86 μsec	60.948 msec	3.72 μsec	1.109 hrs

NOTE:
Input frequency to timers is 1.0752 MHz (default configuration).

Interfaces

MULTIBUS—All signals TTL compatible

iSBX Bus—All signals TTL compatible

Parallel I/O—All signals TTL compatible

Serial I/O—RS232C compatible, configurable as a data set or data terminal

Timer—All signals TTL compatible

Interrupt Requests—All TTL compatible

System Clock (8085A-2 CPU)

4.84 or 2.42 MHz ±0.1% (jumper selectable)

Auxiliary Power

An auxiliary power bus is provided to allow separate power to RAM for systems requiring battery backup of read/write memory. Selection of this auxiliary RAM power bus is made via jumpers on the board.

Connectors

Interface	Double-Sided Pins (qty)	Centers (in.)	Mating Connectors*
MULTIBUS System Bus	86	0.156	ELFAB BS1562043PBB Viking 2KH43/9AMK12 Soldered PCB Mount EDAC 337086540201 ELFAB BW1562D43PBB EDAC 337086540202 ELFAB BW1562A43PBB Wire Wrap
Auxiliary Bus	60	0.100	EDAC 345060524802 ELFAB BS1020A30PBB EDAC 345060540201 ELFAB BW1020D30PBB Wire Wrap
iSBX Bus (2)	36	0.100	iSBX 960-5
Parallel I/O (2)	50	0.100	3M 3415-001 Flat Crimp GTE Sylvania 6AD01251A1DD Soldered
Serial I/O	26	0.100	AMP 15837151 EDAC 345026520202 PCB Soldered 3M 3462-0001 AMP 88373-5 Flat Crimp

***Note:** Connectors compatible with those listed may also be used.

Memory Protect

An active-low TTL compatible memory protect signal is brought out on the auxiliary connector which, when asserted, disables read/write access to RAM memory on the board. This input is provided for the protection of RAM contents during system power-down sequences.

Line Drivers and Terminators

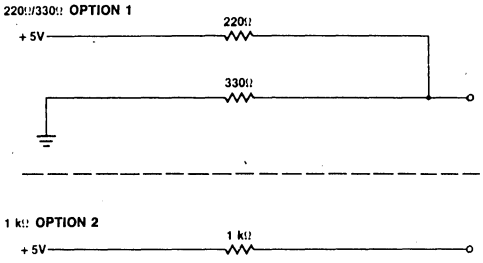
I/O Drivers—The following line drivers and terminators are all compatible with the I/O driver sockets on the iSBC 80/24 Board:

Driver	Characteristic	Sink Current (mA)
7438	I, OC	48
7437	I	48
7432	NI	16
7426	I, OC	16
7409	NI, OC	16
7408	NI	16
7403	I, OC	16
7400	I	16

NOTE:
I = inverting; NI = non-inverting; OC = open collector.

Ports E4 and E8 have 32 mA totem-pole drivers and 1K terminators.

I/O Terminators—220Ω/330Ω divider or 1 kΩ pullup



Bus Drivers

Function	Characteristic	Sink Current (mA)
Data	Tri-State	32
Address	Tri-State	32
Commands	Tri-State	32

Physical Characteristics

Width—12.00 in. (30.48 cm)
Height—6.75 in. (17.15 cm)
Depth—0.50 in. (1.27 cm)
Weight—12.64 oz. (354 gm)

Electrical Characteristics

DC Power Requirements

Configuration	Current Requirements			
	V_{CC} = +5V ±5% (max)	V_{DD} = +12V ±5% (max)	V_{BB} = −5V ±5% (max)	V_{AA} = −12V ±5% (max)
Without EPROM[1]	3.34A	40 mA	—	20 mA
RAM Only[2]	0.14A	—	—	—
With iSBC 530[3]	3.34A	140 mA	—	120 mA
With 4K EPROM[4] (using 2708)	3.74A	300 mA	180 mA	20 mA
With 4K EPROM[4] (using 2758)	4.43A	40 mA	—	20 mA
With 8K EPROM[4] (using 2716)	4.43A	40 mA	—	20 mA
With 16K EPROM[4] (using 2732)	4.71A	40 mA	—	20 mA
With 32K EPROM[4] (using 2764)	4.71A	40 mA	—	20 mA

NOTES:
1. Does not include power for optional EPROM, I/O drivers, and I/O terminators.
2. RAM chips powered via auxiliary power bus.
3. Does not include power for optional EPROM, I/O drivers, and I/O terminators. Power for iSBC 530 Adapter is supplied via serial port connector.
4. Includes power required for four EPROM chips, and I/O terminators installed for 16 I/O lines; all terminator inputs low.

Environmental Characteristics

Operating Temperature—0°C to 55°C

Reference Manual

142648-001—iSBC 80/24 Single Board Computer Hardware Reference Manual (NOT SUPPLIED)

Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

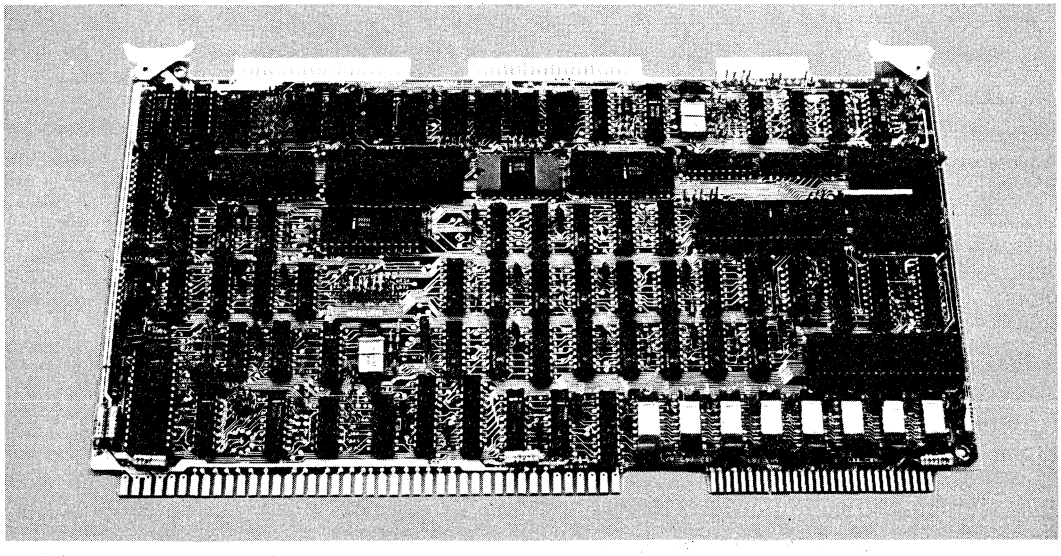
ORDERING INFORMATION

Part Number	Description
SBC 80/24	Single Board Computer

iSBC® 80/30 (or pSBC 80/30*) SINGLE BOARD COMPUTER

- 8085A CPU used as central processing unit
- 16K bytes of dual port dynamic read/write memory with on-board refresh
- Sockets for up to 8K bytes of read only memory
- Sockets for 8041A/8741A Universal Peripheral Interface and interchangeable line drivers and line terminators
- 24 programmable parallel I/O lines with sockets for interchangeable line drivers and terminators
- Programmable synchronous/asynchronous RS232C compatible serial interface with fully software selectable baud rate generation
- Full MULTIBUS® control logic allowing up to 16 masters to share the system
- 12 levels of programmable interrupt control
- Two programmable 16-bit BCD or binary counters
- Auxiliary power bus, memory protect, and power-fail interrupt control logic for RAM battery backup
- Compatible with optional iSBC® 80 CPU, memory, and I/O expansion boards

The iSBC 80/30 Single Board Computer is a member of Intel's complete line of OEM computer systems which take full advantage of Intel's LSI technology to provide economical self-contained computer-based solutions for OEM applications. The iSBC 80/30 is a complete computer system on a single 6.75 × 12.00-inch printed circuit card. The CPU, system clock, read/write memory, nonvolatile read only memory, universal peripheral interface capability, I/O ports and drivers, serial communications interface, priority interrupt logic, programmable timers, MULTIBUS control logic, and bus expansion drivers all reside on the board.



*Same product, manufactured by Intel Puerto Rico, Inc.

October, 1984
AFN-00263B

FUNCTIONAL DESCRIPTION

Central Processing Unit

Intel's powerful 8-bit n-channel 8085A CPU, fabricated on a single LSI chip, is the central processor for the iSBC 80/30. The 8085A CPU is directly software compatible with the Intel 8080A CPU. The 8085A contains six 8-bit general purpose registers and an accumulator. The six general purpose registers may be addressed individually or in pairs, providing both single and double precision operators. The minimum instruction execution time is 1.45 microseconds. The 8085A CPU has a 16-bit program counter. An external stack, located within any portion of iSBC 80/30 read/write memory, may be used as a last-in/first-out storage area for the contents of the program counter, flags, accumulator, and all of the six general purpose registers. A 16-bit stack pointer controls the addressing of this external stack. This stack provides subroutine nesting bounded only by memory size.

Bus Structure

The iSBC 80/30 has an internal bus for all on-board memory and I/O operations and a system bus (i.e., the MULTIBUS) for all external memory and I/O operations. Hence, local (on-board) operations do not tie up the system bus, and allow true parallel processing when several bus masters (i.e., DMA devices, other single board computers) are used in a multimaster scheme. A block diagram of the iSBC 80/30 functional components is shown in Figure 1.

RAM Capacity

The iSBC 80/30 contains 16K bytes of dynamic read/write memory using Intel 2117 RAMs. All RAM read and write operations are performed at maximum processor speed. Power for the on-board RAM may be provided on an auxiliary power bus, and memory protect logic is included for RAM battery backup requirements. The iSBC 80/30 contains a dual port controller, which provides dual port capability for the on-board RAM memory. RAM accesses may occur from either the iSBC 80/30 or from any other bus master interfaced via the

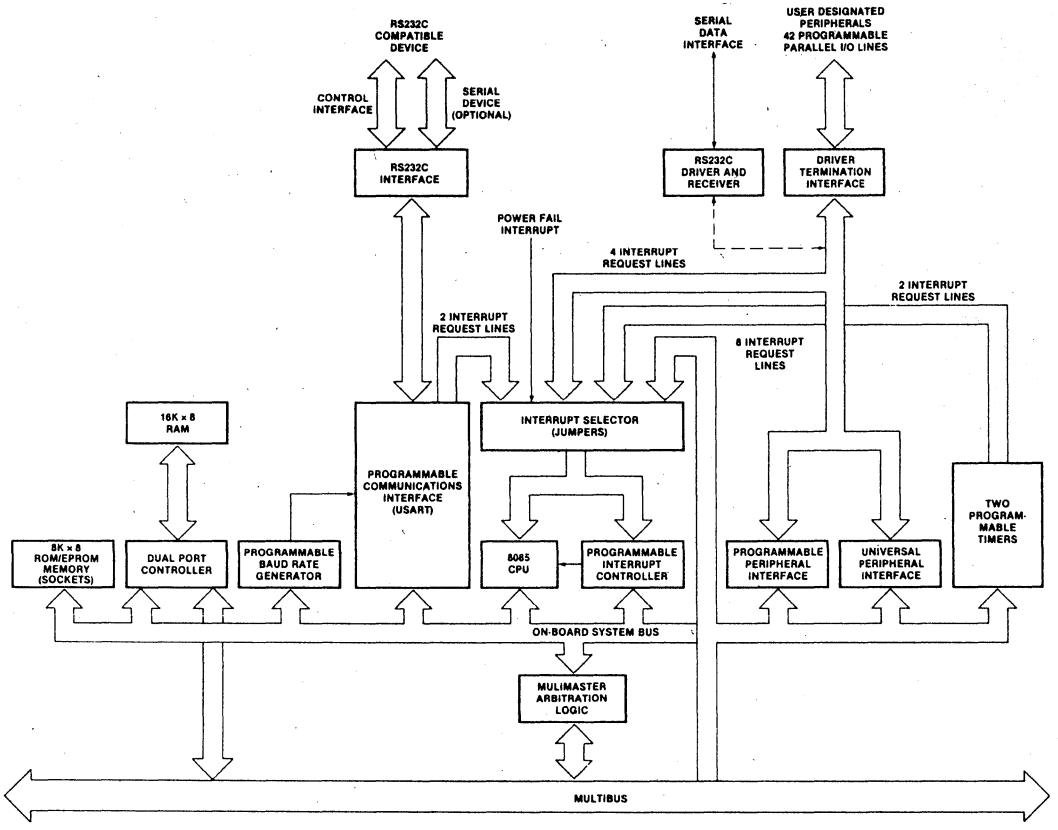


Figure 1. iSBC® 80/30 Single Board Computer Block Diagram

MULTIBUS. Since on-board RAM accesses do not require the MULTIBUS, the bus is available for any other concurrent operations (e.g., DMA data transfers) requiring the use of the MULTIBUS. Dynamic RAM refresh is accomplished automatically by the iSBC 80/30 for accesses originating from either the CPU or via the MULTIBUS. Memory space assignment can be selected independently for on-board and MULTIBUS RAM accesses. The on-board RAM, as seen by the 8085A CPU, may be placed anywhere within the 0- to 64K-address space. The iSBC 80/30 provides extended addressing jumpers to allow the on-board RAM to reside within a one megabyte address space when accessed via the MULTIBUS. In addition, jumper options are provided which allow the user to reserve 8K- and 16K-byte segments of on-board RAM for use by the 8085A CPU only. This reserved RAM space is not accessible via the MULTIBUS and does not occupy any system address space.

EPROM/ROM Capacity

Sockets for up to 8K bytes of nonvolatile read only memory are provided on the iSBC 80/30 board. Read only memory may be added in 1K-byte increments up to a maximum of 2K bytes using Intel 2708 or 2758 erasable and electrically reprogrammable ROMs (EPROMs); in 2K-byte increments up to a maximum of 4K bytes using Intel 2716 EPROMs; or in 4K-byte increments up to 8K bytes maximum using Intel 2732 EPROMs. All on-board EPROM/ROM operations are performed at maximum processor speed.

Parallel I/O Interface

The iSBC 80/30 contains 24 programmable parallel I/O lines implemented using the Intel 8255A Programmable Peripheral Interface. The system software is used to configure the I/O lines in any combination of unidirectional input/output and bidirectional ports indicated in Table 1. Therefore, the I/O interface may be customized to meet specific peripheral requirements. In order to take full advantage of the large number of possible I/O configurations, sockets are provided for interchangeable I/O line drivers and terminators. Hence, the flexibility of the I/O interface is further enhanced by the capability of selecting the appropriate combination of optional line drivers and terminators to provide the required sink current, polarity, and drive/termination characteristics for each application. The 24 programmable I/O lines and signal ground lines are brought out to a 50-pin edge connector that mates with flat, woven, or round cable.

Universal Peripheral Interface (UPI)

The iSBC 80/30 provides sockets for a user supplied Intel 8041A/8741A Universal Peripheral Interface (UPI) chip and the associated line drivers and terminators for the UPI's I/O ports. The 8041A/8741A is a single chip microcomputer containing a CPU, 1K bytes of ROM (8041A) or EPROM (8741A), 64 bytes of RAM, 18 programmable I/O lines, and an 8-bit timer. Special interface registers included in the chip allow the 8041A to function as a slave processor to the iSBC 80/30's 8085A CPU. The UPI allows the user to specify algorithms for controlling user peripherals directly in the chip, thereby relieving the 8085A for other system functions. The iSBC 80/30 provides an RS232C driver and an RS232C receiver for optional connection to the 8041A/8741A in applications where the UPI is programmed to handle simple serial interfaces. For additional information, including 8041A/8741A instructions, refer to the UPI-41A User's Manual and application note AP-41.

Serial I/O

A programmable communications interface using the Intel 8251A Universal Synchronous/Asynchronous Receiver/Transmitter (USART) is contained on the iSBC 80/30. A software selectable baud rate generator provides the USART with all common communication frequencies. The USART can be programmed by the system software to select the desired asynchronous or synchronous serial data transmission technique (including IBM By-Sync). The mode of operation (i.e., synchronous or asynchronous), data format, control character format, parity, and baud rate are all under program control. The 8251A provides full duplex, double buffered transmit and receive capability. Parity, overrun, and framing error detection are all incorporated in the

Table 1. Input/Output Port Modes of Operation

Port	Lines (qty)	Mode of Operation					
		Unidirectional				Bidirectional	Control
		Input		Output			
		Unlatched	Latched & Strobed	Latched	Latched & Strobed		
1	8	X	X	X	X	X	
2	8	X	X	X	X		
3	4	X		X			X[1]
	4	X		X			X[1]

Note

1. Part of port 3 must be used as a control port when either port 1 or port 2 are used as a latched and strobed input or a latched and strobed output port or port 1 is used as a bidirectional port.

USART. The RS232C compatible interface on each board, in conjunction with the USART, provides a direct interface to RS232C compatible terminals, cassettes, and asynchronous and synchronous modems. The RS232C command lines, serial data lines, and signal ground line are brought out to a 26-pin edge connector that mates with RS232C compatible flat or round cable.

Multimaster Capability

The iSBC 80/30 is a full computer on a single board with resources capable of supporting a great variety of OEM system requirements. For those applications requiring additional processing capacity and the benefits of multiprocessing (i.e., several CPUs and/or controllers logically sharing system tasks through communication over the system bus), the iSBC 80/30 provides full MULTIBUS arbitration control logic. This control logic allows up to three iSBC 80/30's or other bus masters to share the system bus in serial (daisy chain) priority fashion, and up to 16 masters to share the MULTIBUS with the addition of an external priority network. The MULTIBUS arbitration logic operates synchronously with a MULTIBUS clock (provided by the iSBC 80/30 or optionally connected directly to the MULTIBUS clock) while data is transferred via a handshake between the master and slave modules. This allows different speed controllers to share resources on the same bus, and transfers via the bus proceed asynchronously. Thus, transfer speed is dependent on transmitting and receiving devices only. This design prevents slow master modules from being handicapped in their attempts to gain control of the bus, but does not restrict the speed at which faster modules can transfer data via the same bus. The most obvious applications for the master-slave capabilities of the bus are multiprocessor configurations, high speed direct memory access (DMA) operations, and high speed peripheral control, but are by no means limited to these three.

Programmable Timers

The iSBC 80/30 provides three independent, fully programmable 16-bit interval timers/event counters utilizing the Intel 8253 Programmable Interval Timer. Each counter is capable of operating in either BCD or binary modes. Two of these timers/counters are available to the systems designer to generate accurate time intervals under software control. Routing for the outputs and gate/trigger inputs of two of these counters is jumper selectable. The outputs may be independently routed to the 8259A Programmable Interrupt Controller, to the I/O line drivers associated with the 8255A Programmable Peripheral Interface, and to the 8041A/8741A Universal Programmable Interface, or may be routed as inputs to the 8255A and 8041A/8741A chips. The gate/trigger inputs may be routed to I/O terminators associated with the 8255A or as output connections from the 8255A. The third interval timer in the 8253 provides the programmable baud rate generator for the iSBC 80/30 RS232C USART serial port. In utilizing the iSBC 80/30, the systems designer simply configures, via software, each timer independently to meet system requirements. Whenever a given time delay or count is needed, software commands to the programmable timers/event counters select the desired function. Seven functions are available, as shown in Table 2. The contents of each counter may be read at any time during system operation with simple read operations for event counting applications, and special commands are included so that the contents of each counter can be read "on the fly".

Table 2. Programmable Timer Functions

Function	Operation
Interrupt on terminal count	When terminal count is reached, an interrupt request is generated. This function is extremely useful for generation of real-time clocks.
Programmable one-shot	Output goes low upon receipt of an external trigger edge or software command and returns high when terminal count is reached. This function is retriggerable.
Rate generator	Divide by N counter. The output will go low for one input clock cycle, and the period from one low-going pulse to the next is N times the input clock period.
Square-wave rate generator	Output will remain high until one-half the count has been completed, and go low for the other half of the count.
Software triggered strobe	Output remains high until software loads count (N). N counts after count is loaded, output goes low for one input clock period.
Hardware triggered strobe	Output goes low for one clock period N counts after rising edge on counter trigger input. The counter is retriggerable.
Event counter	On a jumper selectable basis, the clock input becomes an input from the external system. CPU may read the number of events occurring after the counting "window" has been enabled or an interrupt may be generated after N events occur in the system.

Interrupt Capability

The iSBC 80/30 provides vectoring for 12 interrupt levels. Four of these levels are handled directly by the interrupt processing capability of the 8085A CPU and represent the four highest priority interrupts of the iSBC 80/30. Requests are routed to the 8085A interrupt inputs, TRAP, RST 7.5, RST 6.5, and RST 5.5 (in decreasing order of priority) and each input generates a unique memory address (TRAP: 24H; RST 7.5: 3CH; RST 6.5: 34H; and RST 5.5: 2CH). An 8085A jump instruction at each of these addresses then provides linkage to interrupt ser-

vice routines located independently anywhere in memory. All interrupt inputs with the exception of the trap interrupt may be masked via software. The trap interrupt should be used for conditions such as power-down sequences which require immediate attention by the 8085A CPU. The Intel 8259A Programmable Interrupt Controller (PIC) provides vectoring for the next eight interrupt levels. As shown in Table 3, a selection of four priority processing modes is available to the systems designer for use in designing request processing configurations to match system requirements. Operating mode and priority assignments may be reconfigured dynamically via software at any time during system operation. The PIC accepts interrupt requests from the programmable parallel and serial I/O interfaces, the programmable timers, the system bus, or directly from peripheral equipment. The PIC then determines which of the incoming requests is of the highest priority, determines whether this request is of higher priority than the level currently being serviced, and, if appropriate, issues an interrupt to the CPU. Any combination of interrupt levels may be masked, via software, by storing a single byte in the interrupt mask register of the PIC. The PIC generates a unique memory address for each interrupt level. These addresses are equally spaced at intervals of 4 or 8 (software selectable) bytes. This 32- or 64-byte block may be located to begin at any 32- or 64-byte boundary in the 65,536-byte memory space. A single 8085A jump instruction at each of these addresses then provides linkage to locate each interrupt service routine independently anywhere in memory.

Table 3. Programmable Interrupt Modes

Mode	Operation
Fully nested	Interrupt request line priorities fixed at 0 as highest, 7 as lowest.
Auto-rotating	Equal priority. Each level, after receiving service, becomes the lowest priority level until next interrupt occurs.
Specific priority	System software assigns lowest priority level. Priority of all other levels based in sequence numerically on this assignment.
Polled	System software examines priority-encoded system interrupt status via interrupt status register.

Interrupt Request Generation — Interrupt requests may originate from 18 sources. Two jumper selectable interrupt requests can be automatically generated by the programmable peripheral interface when a byte of information is ready to be transferred to the CPU (i.e., input buffer is full) or a byte of information has been transferred to a peripheral device (i.e., output buffer is empty). Two jumper selectable interrupt requests can be automatically generated by the USART when a character is ready to be transferred to the CPU (i.e., receive channel buffer is full), or a character is ready to be transmitted (i.e., transmit channel data buffer is empty). A jumper selectable request can be generated by each of the programmable timers and by the universal peripheral interface, eight additional interrupt request lines are available to the user for direct interface to user designated peripheral devices via the system bus, and two interrupt request lines may be jumper routed directly from peripherals via the parallel I/O driver/terminator section.

Power-Fail Control

Control logic is also included to accept a power-fail interrupt in conjunction with the AC-low signal from the iSBC 635 Power Supply or equivalent.

Expansion Capabilities

Memory and I/O capacity may be expanded and additional functions added by using Intel MULTIBUS compatible expansion boards. High speed integer and floating point arithmetic capabilities may be added by using the iSBC 310A High Speed Mathematics Unit. Memory may be expanded to 65,536 bytes by adding user specified combinations of RAM boards, EPROM boards, or combination boards. Input/output capacity may be increased by adding digital I/O and analog I/O expansion boards. Mass storage capability may be achieved by adding single or double density diskette controllers as subsystems. Modular expandable backplanes and cardcages are available to support multi-board systems.

System Development Capability

The development cycle of iSBC 80/30-based products may be significantly reduced using Intel's system development tools available today. For those not requiring hardware emulation capability, Intel provides a new low cost microcomputer development system. The iPDS, Personal Development System, provides low cost system development for the iSBC 80/30 board, while at the same time providing personal computer capability for the engineer. The Intellec Series II family of compatible microcomputer development systems provides a range of capability from a low cost disk-based edit debug workstation to a high performance, fully compatible hard-disk-based software development system. A unique in-circuit emulator (ICE-85A) option provides the capability of developing and debugging software directly on the iSBC 80/30 board.

Programming Capability

PL/M-80 — Intel's high level programming language, PL/M, is also available as a resident Intellec microcomputer development system option. PL/M provides the capability to program in a natural, algorithmic language and eliminates the need to manage register usage or

allocate memory. PL/M programs can be written in a much shorter time than assembly language programs for a given application.

FORTRAN-80 — For applications requiring computational and formatted I/O capabilities, the high level FORTRAN-80 programming language is also available as a resident option of the intellec system. FORTRAN-80 meets and exceeds the ANS FORTRAN 77 subset language specification. The FORTRAN-80 compiler produces relocatable object code that may be easily linked with other FORTRAN-80, PL/M, or assembly language program modules. This gives the user wide flexibility in developing software by using the best software tool for a particular functional module within the user's application.

BASIC-80 — A high level language interpreter with extended disk capabilities which operates under the iRMX 80 Real-Time Multi-tasking Executive and translates BASIC-80 source programs into an internally executable form. This language interpreter, provided as a set of linkable object modules, is ideally suited to the OEM who requires a pass thru programming language. The BASIC-80 programs may be created, stored and interpreted on the iSBC 80-based system. The BASIC-80 language has a rich complement of statements, functions, and commands to program applications requiring a full range of 1) string manipulation and disk I/O for data processing, 2) single and double precision floating point and array handling for numeric analysis, or 3) port I/O with mask operations controlled through bit-wise Boolean logical operators.

SPECIFICATIONS

Word Size

Instruction — 8, 16, or 24 bits
Data — 8 bits

Cycle Time

Basic Instruction Cycle — 1.45 μs

Note
Basic instruction cycle is defined as the fastest instruction (i.e., four clock cycles).

Memory Addressing

On-Board ROM/EPROM — 0-07FF (using 2708 or 2758 EPROMs); 0-0FFF (using 2716 EPROMs); 0-1FFF (using 2716 EPROMs; 0-1FFF (using 2732 EPROMs).

On-Board RAM — 16K bytes of dual port RAM starting on a 16K boundary. One or two 8K-byte segments may be reserved for CPU use only.

Memory Capacity

On-Board Read Only Memory — 8K bytes (sockets only)

On-Board RAM — 16K bytes

Off-Board Expansion — Up to 65,536 bytes in user specified combinations of RAM, ROM, and EPROM

Note
Read only memory may be added in 1K, 2K, or 4K-byte increments.

I/O Addressing

On-Board Programmable I/O (see Table 1)

Port	8255A				8041A/8741A		USART	
	1	2	3	Control	Data	Control	Data	Control
Address	E8	E9	EA	EB	E4 or E6	E5 or E7	EC	ED

I/O Capacity

Parallel — 42 programmable lines using one 8255A (24 I/O lines) and an optional 8041A/8741A (18 I/O lines)
Serial — 2 programmable lines using one 8251A and an optional 8041A/8741A programmed for serial operation

Note:
For additional information on the 8041A/8741A refer to the UPI-41 User's Manual (Publication 9800504).

Serial Communications Characteristics

Synchronous — 5—8 bit characters; internal or external character synchronization; automatic sync insertion.

Asynchronous — 5—8 bit characters; break character generation; 1, 1½, or 2 stop bits; false start bit detection.

Baud Rates

Frequency (kHz) (Software Selectable)	Baud Rate (Hz)		
	Synchronous	Asynchronous	
		÷ 16	÷ 64
153.6	—	9600	2400
76.8	—	4800	1200
38.4	38400	2400	600
19.2	19200	1200	300
9.6	9600	600	150
4.8	4800	300	75
2.4	2400	150	—
1.76	1760	110	—

Note
Frequency selected by I/O write of appropriate 16-bit frequency factor to baud rate register (8253 Timer 2).

Interrupts

Addresses for 8259A Registers (Hex notation, I/O address space)

DA Interrupt request register
DA In-service register
DB Mask register
DA Command register
DB Block address register
DA Status (polling register)

Note
Several registers have the same physical address; sequence of access and one data bit of control word determine which register will respond.

Interrupt Levels routed to 8085A CPU automatically vector the processor to unique memory locations:

Interrupt Input	Memory Address	Priority	Type
TRAP	24	Highest	Non-maskable
RST 7.5	3C	↑	Maskable
RST 6.5	34	↓	Maskable
RST 5.5	2C	Lowest	Maskable

Timers

Register Addresses (Hex notation, I/O address space)

DF Control register

DC Timer 0

DD Timer 1

DE Timer 2

Note
Timer counts loaded as two sequential output operations to same address, as given.

Input Frequencies

Reference: 2.46 MHz ±0.1% (0.041 μs period, nominal); 1.23 MHz ± 0.1%(0.81 μs period, nominal); or 153.60 kHz ± 0.1%(6.51 μs period nominal).

Note
Above frequencies are user selectable

Event Rate: 2.46 MHz max

Note
Maximum rate for external events in event counter function.

Output Frequencies/Timing Intervals

Function	Single Timer/Counter		Dual Timer/Counter (Two Timers Cascaded)	
	Min	Max	Min	Max
Real-time interrupt	1.63 μs	427.1 ms	3.26 μs	466.50 min
Programmable one-shot	1.63 μs	427.1 ms	3.26 μs	466.50 min
Rate generator	2.342 Hz	613.5 kHz	0.000036 Hz	306.8 kHz
Square-wave rate generator	2.342 Hz	613.5 kHz	0.000036 Hz	306.8 kHz
Software triggered strobe	1.63 μs	427.1 ms	3.26 μs	466.50 min
Hardware triggered strobe	1.63 μs	427.1 ms	3.26 μs	466.50 min

Interfaces

MULTIBUS — All signals TTL compatible
Parallel I/O — All signals TTL compatible
Interrupt Requests — All TTL compatible
Timer — All signals TTL compatible
Serial I/O — RS232C compatible, data set configuration

System Clock (8085A CPU)

2.76 MHz ±0.1%

Auxiliary Power

An auxiliary power bus is provided to allow separate power to RAM for systems requiring battery backup of read/write memory. Selection of this auxiliary RAM power bus is made via jumpers on the board.

Connectors

Interface	Pins (qty)	Centers (in.)	Mating Connectors
Bus	86	0.156	Viking 2KH43/9AMK12
Parallel I/O	50	0.1	3M 3415-000
Serial I/O	26	0.1	3M 3462-000

Memory Protect

An active-low TTL compatible memory protect signal is brought out on the auxiliary connector which, when asserted, disables read/write access to RAM memory on the board. This input is provided for the protection of RAM contents during system power-down sequences.

Line Drivers and Terminators

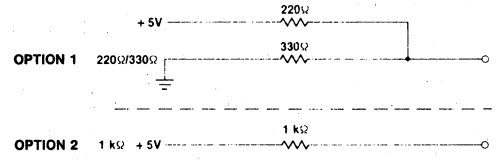
I/O Drivers— The following line drivers are all compatible with the I/O driver sockets on the iSBC 80/30.

Driver	Characteristic	Sink Current (mA)
7438	I,OC	48
7437	I	48
7432	NI	16
7426	I,OC	16
7409	NI,OC	16
7408	NI	16
7403	I,OC	16
7400	I	16

Note
I = inverting; NI = non-inverting; OC = open collector.

Port 1 of the 8255A has 20 mA totem-pole bidirectional drivers and 1 kΩ terminators.

I/O Terminators — 220Ω/330Ω divider or 1 kΩ pullup



Bus Drivers

Function	Characteristic	Sink Current (mA)
Data	Tri-state	50
Address	Tri-state	50
Commands	Tri-state	32

Physical Characteristics

Width — 12.00 in. (30.48 cm)
Height — 6.75 in. (17.15 cm)
Depth — 0.50 in. (1.27 cm)
Weight — 18 oz. (509.6 gm)

Electrical Characteristics

DC Power Requirements

Configuration	Current Requirements			
	V_{CC} = +5V ±5%(max)	V_{DD} = +12V ±5%(max)	V_{BB} = −5V ±5%(max)	V_{AA} = −12V ±5%(max)
Without EPROM[1]	I_{CC} = 3.5A	I_{DD} = 220 mA	I_{BB} = —	I_{AA} = 50 mA
With 8041/8741[2]	3.6A	220 mA	—	50 mA
RAM only[3]	350 mA	20 mA	2.5 mA	—
With iSBC 530[4]	3.5A	320 mA	—	150 mA
With 2K EPROM[5] (using 8708)	4.4A	350 mA	95 mA	40 mA
With 2K EPROM[5] (using 2758)	4.6A	220 mA	—	50 mA
With 4K EPROM[5] (using 2716)	4.6A	220 mA	—	50 mA
With 8K EPROM[5] (using 2332)	4.6A	220 mA	—	50 mA

Notes

1. Does not include power required for optional EPROM/ROM, 8041A/8741A I/O drivers, and I/O terminators.
2. Does not include power required for optional EPROM/ROM, I/O drivers and I/O terminators.
3. RAM chips powered via auxiliary power bus
4. Does not include power required for optional EPROM/ROM, 8041A/8741A I/O drivers, and I/O terminators. Power for iSBC 530 is supplied through the serial port connector.
5. Includes power required for two EPROM/ROM chips, 8041A/8741A and 220Ω/330Ω input terminators installed for 34 I/O lines; all terminator inputs low.

Environmental Characteristics

Operating Temperature — 0°C to 55°C

Reference Manual

9800611B — iSBC 80/30 Single Board Computer Hardware Reference Manual (NOT SUPPLIED)

Reference manuals are shipped with each product only if designated SUPPLIED (see above). Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

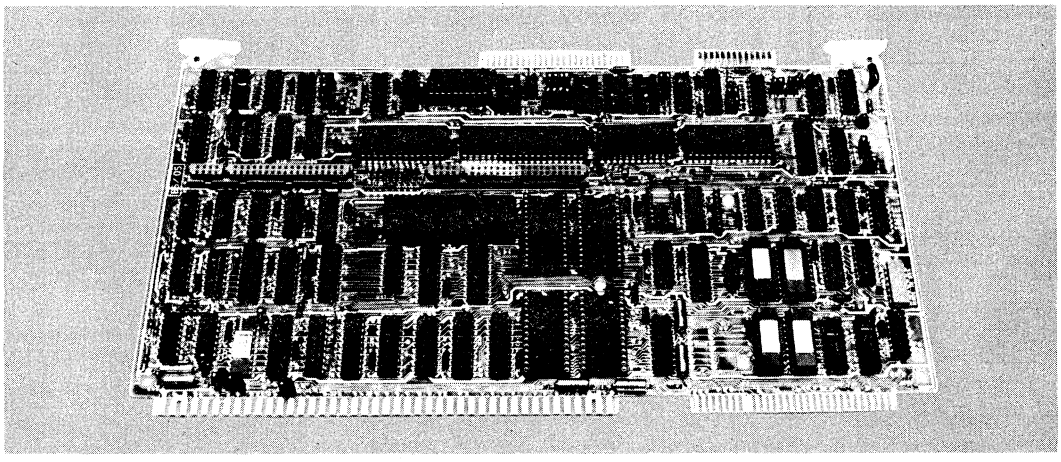
ORDERING INFORMATION

Part Number	Description
SBC 80/30	Single Board Computer with 16K bytes RAM

iSBC® 86/05
SINGLE BOARD COMPUTER

- iAPX 86/10 (8086-2) Microprocessor with 5 or 8 MHz CPU clock
- Fully software compatible with iSBC® 86/12A Single Board Computer
- Optional iAPX 86/20 Numeric Data Processor with iSBC ® 337 MULTIMODULE ™ Processor
- 8K bytes of static RAM; expandable on-board to 16K bytes
- Sockets for up to 64K bytes of JEDEC 24/28-pin standard memory devices; expandable on-board to 128K bytes
- Two iSBX™ bus connectors
- 24 programmable parallel I/O lines
- Programmable synchronous/asynchronous RS232C compatible serial interface with software selectable baud rates
- Two programmable 16-bit BCD or binary timers/event counters
- 9 Levels of vectored interrupt control, expandable to 65 levels
- MULTIBUS® interface for multimaster configurations and system expansion
- Supported by a complete family of single board computers, memory, digital and analog I/O, peripheral controllers, packaging and software

The iSBC 86/05 Single Board Computer is a member of Intel's complete line of OEM microcomputer systems which take full advantage of Intel's technology to provide economical, self-contained, computer-based solutions for OEM applications. The iSBC 86/05 board is a complete computer system on a single 6.75 x 12.00-in. printed circuit card. The CPU, system clock, read/write memory, nonvolatile read only memory, I/O ports and drivers, serial communications interface, priority interrupt logic and programmable timers, all reside on the board. The large control storage capacity makes the iSBC 86/05 board ideally suited for control-oriented applications such as process control, instrumentation, industrial automation, and many others.



OCTOBER 1984
AFN-01819B

FUNCTIONAL DESCRIPTION

Central Processing Unit

The central processor for the iSBC 86/05 board is Intel's iAPX 86/10 (8086-2) CPU. A clock rate of 8 MHz is supported with a jumper selectable option of 5 MHz. The CPU architecture includes four 16-bit byte addressable data registers, two 16-bit memory base pointer registers and two 16-bit index registers, all accessed by a total of 24 operand addressing modes for comprehensive memory addressing and for support of the data structures required for today's structured, high level languages as well as assembly language.

Instruction Set

The 8086 instruction repertoire includes variable length instruction format (including double operand instructions), 8-bit and 16-bit signed and unsigned arithmetic operators for binary, BCD and unpacked ASCII data, and iterative word and byte string manipulation functions.

For enhanced numerics processing capability, the iSBC 337 MULTIMODULE Numeric Data Processor extends the iAPX 86/10 architecture and data set. Over 60 numeric instructions offer arithmetic, trigonometric, transcendental, logarithmic and exponential instructions. Supported data types include 16, 32, and 64-bit integer, and 32 and 64-bit floating point, 18-digit packed BCD and 80-bit temporary.

Architectural Features

A 6-byte instruction queue provides pre-fetching of sequential instructions and can reduce the 750 nsec minimum instruction cycle to 250 nsec for queued instructions. The stack-oriented architecture readily supports modular programming by facilitating fast, simple, inter-module communication, and other programming constructs needed for asynchronous real-time systems. The memory expansion capabilities offer a 1 megabyte addressing range. The dynamic relocation scheme allows ease in segmentation of pure procedure and data for efficient memory utilization. Four segment registers (code, stack, data, extra) contain program loaded offset values which are used to map 16-bit addresses to 20-bit addresses. Each register maps 64K bytes at a time and activation of a specific register is controlled explicitly by program control and is also selected implicitly by specific functions and instructions. All Intel languages support the extended memory capability, relieving the programmer of managing the megabyte memory space, yet allowing explicit control when necessary.

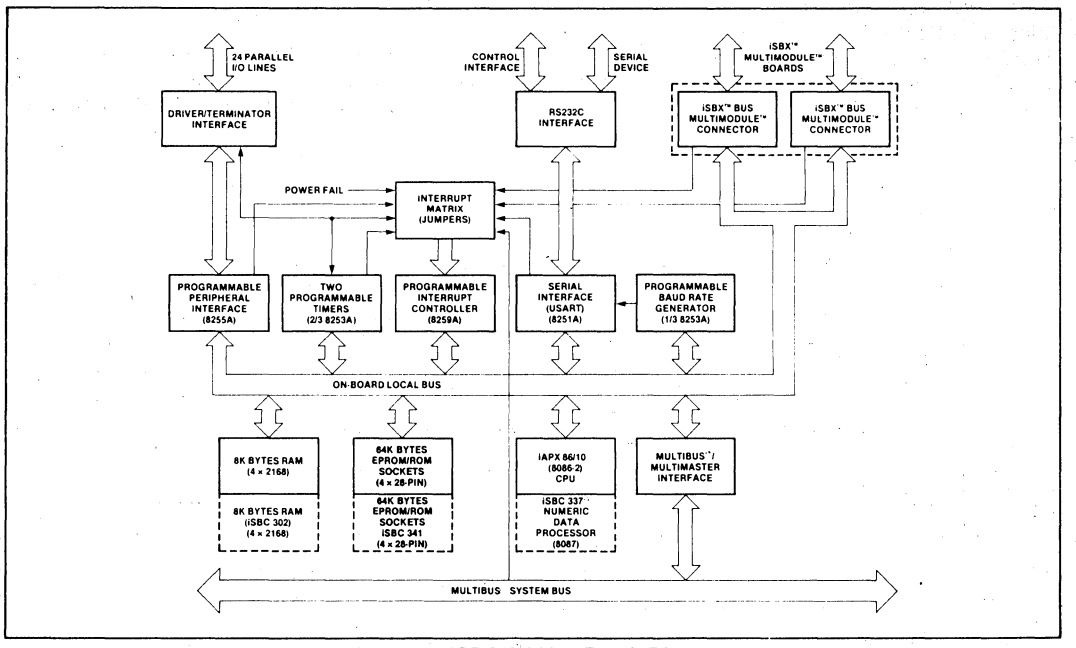


Figure 1. iSBC ® 86/05 Block Diagram

Memory Configuration

The iSBC 86/05 microcomputer contains 8K bytes of high-speed static RAM on-board. In addition, the on-board RAM may be expanded to 16K bytes with the iSBC 302 MULTIMODULE RAM option which mounts on the iSBC 86/05 board. All on-board RAM is accessed by the 8086-2 CPU with no wait states, yielding a memory cycle time of 500 nsec.

In addition to the on-board RAM, the iSBC 86/05 board has four 28-pin sockets, configured to accept JEDEC 24/28-pin standard memory devices. Up to 64K bytes of EPROM are supported in 16K-byte increments with Intel 27128 EPROMs. The iSBC 86/05 board is also compatible with the 2716, 2732A, and 2764 EPROMs offering expansion to 8, 16 and 32K bytes, respectively.

With the addition of the iSBC 341 MULTIMODULE EPROM option, the on-board capacity for these devices is doubled, providing up to 128K bytes of EPROM capacity on-board.

Parallel I/O Interface

The iSBC 86/05 Single Board Computer contains 24 programmable parallel I/O lines implemented using the Intel 8255A Programmable Peripheral Interface. The system software is used to configure the I/O lines in any combination of unidirectional input/output and bidirectional ports indicated in Table 1. In order to take advantage of the large number of possible I/O configurations, sockets are provided for interchangeable I/O line drivers and terminators, allowing the selection of the appropriate combination of optional line drivers and terminators with the required drive/termination characteristics. The 24 programmable I/O lines and signal ground lines are brought out to a 50-pin edge connector.

Serial I/O

A programmable communications interface using the Intel 8251A Universal Synchronous/Asynchronous Receiver/Transmitter (USART) is contained on the iSBC 86/05 board. A software selectable baud rate generator provides the USART with all common communication frequencies. The mode of operation (i.e., synchronous or asynchronous), data format, control character format, parity, and baud rate are all under program control. The 8251A provides full duplex, double buffered transmit and receive capability. Parity, overrun, and framing error detection are all incorporated in the USART. The RS232C compatible interface on each board, in conjunction with the USART, provides a direct interface to RS232C compatible terminals, cassettes, and asynchronous and synchronous modems. The RS232C command lines, serial data lines and signal ground line are brought out to a 26-pin edge connector.

Programmable Timers

The iSBC 86/05 board provides three independent, fully programmable 16-bit interval timers/event counters utilizing the Intel 8253 Programmable

Table 1. Input/Output Port Modes of Operation

Port	Lines (qty)	Mode of Operation					Control
		Unidirectional				Bidirectional	
		Input		Output			
		Latched	Latched & Strobed	Latched	Latched & Strobed		
1	8	X	X	X	X	X	
2	8	X	X	X	X		
3	4	X		X			X[1]
	4	X		X			X[1]

NOTE:
1. Part of port 3 must be used as a control port when either port 1 or port 2 are used as a latched and strobed input or a latched and strobed output port or port 1 is used as a bidirectional port.

Interval Timer. Each counter is capable of operating in either BCD or binary modes. Two of these timers/counters are available to the systems designer to generate accurate time intervals under software control. Routing for the outputs and gate/trigger inputs of two of these counters is jumper selectable. The outputs may be independently routed to the 8259A Programmable Interrupt Controller and to the I/O terminators associated with the 8255A to allow external devices or an 8255A port to gate the timer or to count external events. The third interval timer in the 8253 provides the programmable baud rate generator for the iSBC 86/05 board RS232C USART serial port. The system software configures each timer independently to select the desired function. Seven functions are available as shown in Table 2. The contents of each counter may be read at any time during system operation.

Table 2. Programmable Timer Functions

Function	Operation
Interrupt on terminal count	When terminal count is reached, an interrupt request is generated. This function is extremely useful for generation of real-time clocks.
Programmable one-shot	Output goes low upon receipt of an external trigger edge or software command and returns high when terminal count is reached. This function is retriggerable.
Rate generator	Divide by N counter. The output will go low for one input clock cycle, and the period from one low going pulse to the next is N times the input clock period.
Square-wave rate generator	Output will remain high until one-half the count has been completed, and go low for the other half of the count.
Software triggered strobe	Output remains high until software loads count (N). N counts after count is loaded, output goes low for one input clock period.
Hardware triggered strobe	Output goes low for one clock period N counts after rising edge counter trigger input. The counter is retriggerable.
Event counter	On a jumper selectable basis, the clock input becomes an input from the external system. CPU may read the number of events occurring after the counter "window" has been enabled or an interrupt may be generated after N events occur in the system.

iSBX™ MULTIMODULE™On-Board Expansion

Two 8/16-bit iSBX MULTIMODULE connectors are provided on the iSBC 86/05 microcomputer. Through these connectors, additional on-board I/O functions may be added. iSBX MULTIMODULES optimally support functions provided by VLSI peripheral components such as additional parallel and serial I/O, analog I/O, small mass storage device controllers (e.g., cassettes and floppy disks), and other custom interfaces to meet specific needs. By mounting directly on the single board computer, less interface logic, less power, simpler packaging, higher performance, and lower cost result when compared to other alternatives such as MULTIBUS form factor compatible boards. The iSBX connectors on the iSBC 86/05 provide all signals necessary to interface to the local on-board bus, including 16 data lines for maximum data transfer rates. iSBX MULTIMODULE boards designed with 8-bit data paths and using the 8-bit iSBX connector are also supported on the iSBC 86/05 microcomputer. A broad range of iSBX MULTIMODULE options are available in this family from Intel. Custom iSBX modules may also be designed for use on the iSBC 86/05 board. An iSBX bus interface specification and iSBX connectors are available from Intel.

MULTIBUS® SYSTEM BUS AND MULTIMASTER CAPABILITIES

Overview

The MULTIBUS system bus is Intel's industry standard microcomputer bus structure. Both 8 and 16-bit single board computers are supported on the MULTIBUS structure with 24 address and 16 data lines. In its simplest application, the MULTIBUS system bus allows expansion of functions already contained on a single board computer (e.g., memory and digital I/O). However, the MULTIBUS structure also allows very powerful distributed processing configurations with multiple processors and intelligent slave I/O, and peripheral boards capable of solving the most demanding microcomputer applications. The MULTIBUS system bus is supported with a broad array of board level products, LSI interface components, detailed published specifications and application notes.

Expansion Capabilities

Memory and I/O capacity may be expanded and additional functions added using Intel MULTIBUS

compatible expansion boards. Memory may be expanded by adding user specified combinations of RAM boards, EPROM boards, or combination boards. Input/output capacity may be added with digital I/O and analog I/O expansion boards. Mass storage capability may be achieved by adding single or double density diskette controllers, or hard disk controllers. Modular expandable backplanes and cardcages are available to support multiboard systems.

Multimaster Capabilities

For those applications requiring additional processing capacity and the benefits of multiprocessing (i.e., several CPUs and/or controllers logically sharing system tasks through communication of the system bus), the iSBC 86/05 board provides full MULTIBUS arbitration control logic. This control logic allows up to three iSBC 86/05 boards or other bus masters, including iSBC 80 family MULTIBUS compatible 8-bit single board computers to share the system bus using a serial (daisy chain) priority scheme and allows up to 16 masters to share the MULTIBUS system bus with an external parallel priority decoder. In addition to the multiprocessing configurations made possible with multimaster capability, it also provides a very efficient mechanism for all forms of DMA (Direct Memory Access) transfers.

Interrupt Capability

The iSBC 86/05 board provides 9 vectored interrupt levels. The highest level is the NMI (Non-Maskable Interrupt) line which is directly tied to the 8086 CPU. This interrupt is typically used for signaling catastrophic events (e.g., power failure). The Intel 8259A Programmable Interrupt Controller (PIC) provides control and vectoring for the next eight interrupt levels. As shown in Table 3, a selection of four priority processing modes is available for use in designing request processing configurations to match system requirements for efficient interrupt servicing with minimal latencies. Operating mode and priority assignments may be reconfigured dynamically via software at any time during system operation. The PIC accepts interrupt requests from all on-board I/O resources and from the MULTIBUS system bus. The PIC then resolves requests according to the selected mode and, if appropriate, issues an interrupt to the CPU. Any combination of interrupt levels may be masked via software, by storing a single byte in the interrupt mask register of the PIC. In systems requiring additional interrupt levels, slave 8259A PICs may be interfaced via the MULTIBUS system bus, to generate additional vector addresses, yielding a total of 65 unique interrupt levels.

Table 3. Programmable Interrupt Modes

Mode	Operation
Fully nested	Interrupt request line priorities fixed at 0 as highest, 7 as lowest.
Auto-rotating	Equal priority. Each level, after receiving service, becomes the lowest priority level until next interrupt occurs.
Specific priority	System software assigns lowest priority level. Priority of all other levels based in sequence numerically on this assignment.
Polled	System software examines priority-encoded system interrupt status via interrupt status register.

Interrupt Request Generation

Interrupt requests to be serviced by the iSBC 86/05 board may originate from 24 sources. Table 4 includes a list of devices and functions supported by interrupts. All interrupt signals are brought to the interrupt jumper matrix where any combination of interrupt sources may be strapped to the desired interrupt request level on the 8259A PIC or the NMI input to the CPU directly.

Power-Fail Control and Auxiliary Power

Control logic is also included to accept a power-fail interrupt in conjunction with the AC-low signal from the iSBC 635 and iSBC 640 Power Supply or equivalent, to initiate an orderly shut down of the system in the event of a power failure. Additionally, an active-low TTL compatible memory protect signal is brought out on the auxiliary connector which, when asserted, disables read/write access to RAM memory on the board. This input is provided for the protection of RAM contents during system power-down sequences. An auxiliary power bus is also provided to allow separate power to RAM for systems requiring battery back-up of read/write memory. Selection of this auxiliary RAM power bus is made via jumpers on the board.

System Development Capabilities

The development cycle of iSBC 86/05 products can be significantly reduced and simplified by

using the Intellec Series Microcomputer Development Systems. The Assembler, Locating Linker, Library Manager, Text Editor and System Monitor are all supported by the ISIS-II disk-based operating system. To facilitate conversion of 8080A/8085A assembly language programs to run on the iSBC 86/05 board, CONV-86 is available under the ISIS-II operating system.

IN-CIRCUIT EMULATOR

The ICE-86 In-Circuit Emulator provides the necessary link between the software development environment provided by the Intellec system and the "target" iSBC 86/05 execution system. In addition to providing the mechanism for loading executable code and data into the iSBC 86/05 board, the ICE-86 In-Circuit Emulator provides a sophisticated command set to assist in debugging software and final integration of the user hardware and software.

PL/M-86

Intel's system's implementation language, PL/M-86, is also available as an Intellec Microcomputer Development System option. PL/M-86 provides the capability to program in algorithmic language and eliminates the need to manage register usage or allocate memory while still allowing explicit control of the system's resources when needed.

Run-Time Support

Intel also offers two run-time support packages; iRMX 88 Realtime Multitasking Executive and the iRMX 86 Operating System. iRMX 88 is a simple, highly configurable and efficient foundation for small, high performance applications. Its multitasking structure establishes a solid foundation for modular system design and provides task scheduling and management, intertask communication and synchronization, and interrupt servicing for a variety of peripheral devices. Other configurable options include terminal handlers, disk file system, debuggers and other utilities. iRMX 86 is a high functional operating system with a very rich set of features and options based on an object-oriented architecture. In addition to being modular and configurable, functions beyond the nucleus include a sophisticated file management and I/O system, and powerful human interface. Both packages are easily customized and extended by the user to match unique requirements.

Table 4. Interrupt Request Sources

Device	Function	Number of Interrupts
MULTIBUS interface	Requests from MULTIBUS resident peripherals or other CPU boards	8; may be expanded to 64 with slave 8259A PICs on MULTIBUS boards
8255A Programmable Peripheral Interface	Signals input buffer full or output buffer empty; also BUS INTR OUT general purpose interrupt from driver/terminator sockets	3
8251A USART	Transmit buffer empty and receive buffer full	2
8253 Timers	Timer 0, 1 outputs; function determined by timer mode	2
iSBX connectors	Function determined by iSBX MULTIMODULE board	4 (2 per iSBX connector)
Bus fail safe timer	Indicates addressed MULTIBUS resident device has not responded to command within 6 msec	1
Power fail interrupt	Indicates AC power is not within tolerance	1
Power line clock	Source of 120 Hz signal from power supply	1
External interrupt	General purpose interrupt from auxiliary (P2) connector on backplane	1
iSBC 337 MULTIMODULE Numeric Data Processor	Indicates error or exception condition	1

SPECIFICATIONS

Word Size

INSTRUCTION — 8, 16, 24, or 32 bits

DATA — 8, 16 bits

System Clock

5.00 MHz or 8.00 MHz ±0.1% (jumper selectable)

Cycle Time

BASIC INSTRUCTION CYCLE

At 8 MHz — 750 nsec
— 250 nsec (assumes instruction in the queue)

At 5 MHz — 1.2 μsec
— 400 nsec (assumes instruction in the queue)

NOTES:
Basic instruction cycle is defined as the fastest instruction time (i.e., two clock cycles).

Memory Cycle Time

RAM — 500 nsec (no wait states)

EPROM — Jumper selectable from 500 nsec to 875 nsec

Memory Capacity/Addressing

ON-BOARD EPROM

Device	Total Capacity	Address Range
2716	8K bytes	$FE000-FFFFF_H$
2732A	16K bytes	$FC000-FFFFF_H$
2764	32K bytes	$F8000-FFFFF_H$
27128	64K bytes	$F0000-FFFFF_H$

WITH iSBC 341 MULTIMODULE EPROM

Device	Total Capacity	Address Range
2716	16K bytes	$FC000-FFFFF_H$
2732A	32K bytes	$F8000-FFFFF_H$
2764	64K bytes	$F0000-FFFFF_H$
27128	128K bytes	$E0000-FFFFF_H$

NOTES:
iSBC 86/05 EPROM sockets support JEDEC 24/28-pin standard EPROMs and RAMs; iSBC 341 sockets also support E^2PROMs.

ON-BOARD RAM

8K bytes — $0-1FFF_H$

WITH iSBC 302 MULTIMODULE RAM

16K bytes — $0-3FFF_H$

I/O Capacity

PARALLEL — 24 programmable lines using one 8255A.

SERIAL — 1 programmable line using one 8251A

iSBX MULTIMODULE — 2 iSBX MULTIMODULE boards

Serial Communications Characteristics

SYNCHRONOUS — 5-8 bit characters; internal or external character synchronization; automatic sync insertion

ASYNCHRONOUS — 5-8 bit characters; break character generation; 1, 1½, or 2 stop bits; false start bit detection

BAUD RATES

Frequency (kHz) (Software Selectable)	Baud Rate (Hz)		
	Synchronous	Asynchronous	
		÷16	÷64
153.6	—	9600	2400
76.8	—	4800	1200
38.4	38400	2400	600
19.2	19200	1200	300
9.6	9600	600	150
4.8	4800	300	75
2.4	2400	150	—
1.76	1760	110	—

NOTES:
Frequency selected by I/O write of appropriate 16-bit frequency factor to baud rate register (8253 Timer 2).

Timers

INPUT FREQUENCIES

Reference: 2.46 MHz ±0.1% (0.041 μsec period, nominal); or 153.60 kHz ±0.1% (6.51 μsec period, nominal)

NOTES:
Above frequencies are user selectable.

Event Rate: 2.46 MHz max

OUTPUT FREQUENCIES/TIMING INTERVALS

Function	Single Timer/Counter		Dual Timer/Counter (Two Timers Cascaded)	
	Min	Max	Min	Max
Real-time Interrupt	1.63 μs	427.1 ms	3.26s	466.50 min
Programmable one-shot	1.63 μs	427.1 ms	3.26s	466.50 min
Rate generator	2.342 Hz	613.5 kHz	0.000036 Hz	306.8 kHz
Square-wave rate generator	2.342 Hz	613.5 kHz	0.000036 Hz	306.8 kHz
Software triggered strobe	1.63 μs	427.1 ms	3.26s	466.50 min
Hardware triggered strobe	1.63 μs	427.1 ms	3.26s	466.50 min
Event counter	—	2.46 MHz	—	—

Interfaces

MULTIBUS — All signals TTL compatible
iSBX BUS — All signals TTL compatible
PARALLEL I/O — All signals TTL compatible
SERIAL I/O — RS232C compatible, configurable as a data set or data terminal
TIMER — All signals TTL compatible
INTERRUPT REQUESTS — All TTL compatible

Connectors

Interface	Double-Sided Pins (qty)	Centers (in.)	Mating Connectors
MULTIBUS™ System	86	0.156	Viking 3KH43/9AMK12 Wire Wrap
iSBX™ Bus			
8-Bit Data	36	0.1	iSBX 960-5
16-Bit Data	44	0.1	iSBX 961-5
Parallel I/O (2)	50	0.1	3M 3415-000 Flat or TI H312125 Pins
Serial I/O	26	0.1	3M 3462-0001 Flat or AMP 88106-1 Flat

Line Drivers and Terminators

I/O DRIVERS — The following line drivers are all compatible with the I/O driver sockets on the iSBC 86/05 board

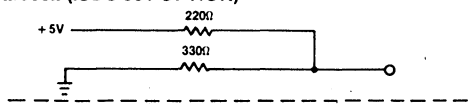
Driver	Characteristic	Sink Current (mA)
7438	I,OC	48
7437	I	48
7432	NI	16
7426	I,OC	16
7409	NI,OC	16
7408	NI	16
7403	I,OC	16
7400	I	16

NOTES:
I = inverting; NI = non-inverting; OC = open collector.

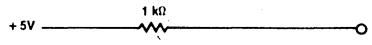
Port 1 of the 8255A has 20 mA totem-pole bidirectional drivers and 1 kΩ terminators

I/O TERMINATORS — 220Ω/330Ω divider or 1 kΩ pullup

220Ω/330Ω (iSBC 901 OPTION)



1 kΩ (iSBC 902 OPTION)



MULTIBUS® Drivers

Function	Characteristic	Sink Current (mA)
Data	Tri-State	50
Address	Tri-State	50
Commands	Tri-State	32
Bus Control	Open Collector	20

Physical Characteristics

WIDTH — 12.00 in. (30.48 cm)
HEIGHT — 6.75 in. (17.15 cm)
DEPTH — 0.70 in. (1.78 cm)
WEIGHT — 14 oz (388 gm)

Electrical Characteristics

DC POWER REQUIREMENTS

Configuration	Current Requirements (All Voltages ±5%)		
	+5V	+12V	−12V
Without EPROM[1]	4.7A	25 mA	23 mA
RAM only[2]	120 mA		
With 8K EPROM[3] (using 2716)	5.0A	25 mA	23 mA
With 16K EPROM[3] (using 2732A)	4.9A	25 mA	23 mA
With 32K EPROM[3] (using 2764)	4.9A	25 mA	23 mA

NOTES:
1. Does not include power for optional ROM/EPROM, I/O drivers, and I/O terminators.
2. RAM chips powered via auxiliary power bus in power-down mode.
3. Includes power required for 4 ROM/EPROM chips, and I/O terminators installed for 16 I/O lines; all terminator inputs low.

Environmental Characteristics

OPERATING TEMPERATURE — 0°C to 55°C
RELATIVE HUMIDITY — to 90% (without condensation)

Reference Manual

143153-001 — iSBC 86/05 Hardware Reference Manual (NOT SUPPLIED)

Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

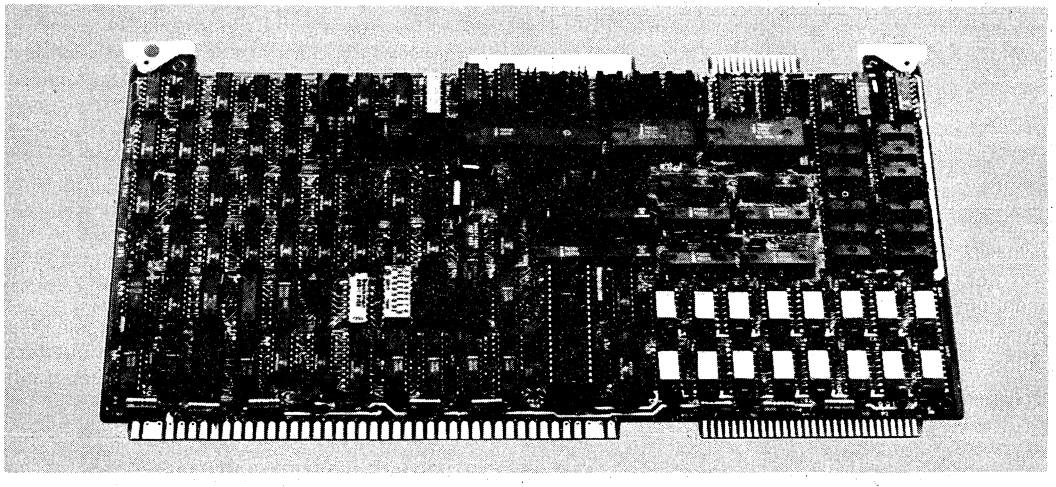
ORDERING INFORMATION

Part Number	Description
SBC 86/05	16-bit Single Board Computer with 8K bytes RAM

iSBC® 86/12A (or pSBC 86/12A*) SINGLE BOARD COMPUTER

- 8086 16-bit HMOS microprocessor central processor unit
- 32K bytes of dual-port read/write memory expandable on-board to 64K bytes with on-board refresh
- Sockets for up to 16K bytes of read only memory expandable on-board to 32K bytes
- System memory expandable to 1 megabyte
- 24 programmable parallel I/O lines with sockets for interchangeable line drivers and terminators
- Programmable synchronous/asynchronous RS232C compatible serial interface with software selectable baud rates
- Two programmable 16-bit BCD or binary timers/event counters
- 9 levels of vectored interrupt control, expandable to 65 levels
- Auxiliary power bus and power fail interrupt control logic for read/write memory battery backup
- MULTIBUS® interface for multimaster configurations and system expansion
- Compatible with iSBC® 337 MULTIMODULE™ Numeric Data Processor
- Compatible with iSBC® 80 family single board computers, memory, digital and analog I/O, and peripheral controller boards

The iSBC 86/12A Single Board Computer is a member of Intel's complete line of OEM microcomputer systems which take full advantage of Intel's LSI technology to provide economical self-contained computer based solutions for OEM applications. The iSBC 86/12A board is a complete computer system on a single 6.75 x 12.00-inch printed circuit card. The CPU, system clock, read/write memory, nonvolatile read only memory, I/O ports and drivers, serial communications interface, priority interrupt logic and programmable timers, all reside on the board. Full MULTIBUS interface logic is included to offer compatibility with the Intel OEM Microcomputer Systems family of Single Board Computers, expansion memory options, digital and analog I/O expansion boards and peripheral controllers.



*Same product, manufactured by Intel Puerto Rico, Inc.

FUNCTIONAL DESCRIPTION

Central Processing Unit

The central processor for the iSBC 86/12A board is Intel's 8086, a powerful 16-bit HMOS device. The 225 sq. mil chip contains 29,000 transistors and has a clock rate of 5MHz. The architecture includes four (4) 16-bit byte addressable data registers, two (2) 16-bit memory base pointer registers and two (2) 16-bit index registers, all accessed by a total of 24 operand addressing modes for complex data handling and very flexible memory addressing.

Instruction Set — The 8086 instruction repertoire includes variable length instruction format (including double operand instructions), 8-bit and 16-bit signed and unsigned arithmetic operators for binary, BCD and unpacked ASCII data, and iterative word and byte string manipulation functions. In addition, the iSBC 337 MULTIMODULE Numeric Data Processor may be installed to add over 60 numeric instructions and hardware support for multiple precision integer and floating point data types.

Architectural Features — A 6-byte instruction queue provides pre-fetching of sequential instructions and can reduce the 1.2μsec minimum instruction cycle to 400 nsec for queued instructions. The stack oriented architecture facilitates nested subroutines and co-routines, reentrant code and powerful interrupt handling. The memory expansion capabilities offer a 1 megabyte addressing range. The dynamic relocation scheme allows ease in segmentation of pure procedure and data for efficient memory utilization. Four segment registers (code, stack, data, extra) contain program loaded offset values which are used to map 16-bit addresses to 20-bit addresses. Each register maps 64K-bytes at a time and activation of a specific register is controlled explicitly by program control and is also selected implicitly by specific functions and instructions.

Bus Structure

The iSBC 86/12A microcomputer has three buses: an internal bus for communicating with on-board memory and I/O options, the MULTIBUS system bus for referencing additional memory and I/O options, and the dual-port bus which allows access to RAM from the on-board CPU and the MULTIBUS system bus. Local (on-board) accesses do not require MULTIBUS communication, making the system bus available for use by other MULTIBUS masters (i.e. DMA devices and other single board computers transferring to additional system memory). This feature allows true parallel processing in a multiprocessor environment. In addition, the MULTIBUS interface can be used for system expansion through the use of other 8- and 16-bit iSBC computers, memory and I/O expansion boards.

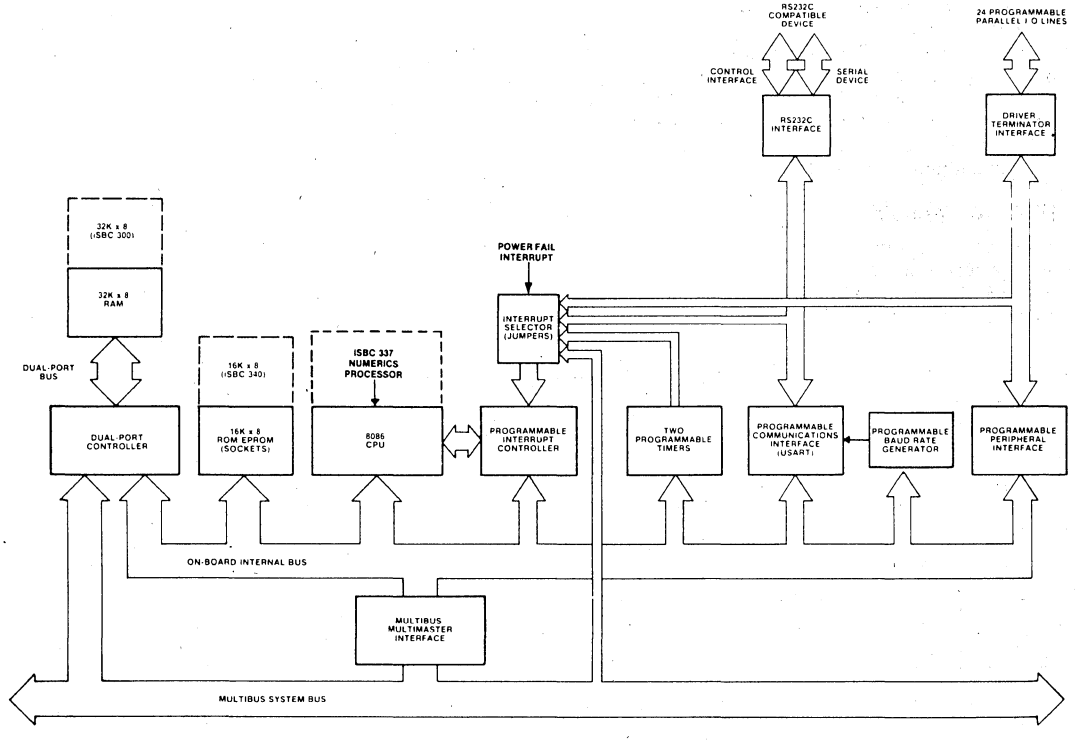


Figure 1. iSBC®86/12A Single Board Computer Block Diagram

RAM Capabilities

The iSBC 86/12A microcomputer contains 32K bytes of dynamic read/write memory using 16K-bit 2117 RAMs. In addition, the on-board RAM complement may be expanded to 64K bytes with the iSBC 300 32K-byte MULTIMODULE RAM option. Power for the on-board RAM and refresh circuitry may be optionally provided on an auxiliary power bus, and memory protect logic is included for RAM battery backup requirements. The iSBC 86/12A board contains a dual-port controller which allows access to the on-board RAM (32K bytes or 64K bytes when the iSBC 300 module is included with the iSBC 86/12A board) from the iSBC 86/12A CPU and from any other MULTIBUS master via the system bus. The dual-port controller allows 8- and 16-bit accesses from the MULTIBUS system bus, and the on-board CPU transfers data to RAM over a 16-bit data path. Priorities have been established such that memory refresh is guaranteed by the on-board refresh logic and that the on-board CPU has priority over MULTIBUS system bus requests for access to RAM. The dual-port controller includes independent addressing logic for RAM access from the on-board CPU and from the MULTIBUS system bus. The on-board CPU will always access RAM starting at location 00000_H. Address jumpers allow on-board RAM to be located starting on any 8K-byte boundary within a 1 megabyte address range for accesses from the MULTIBUS system bus. In conjunction with this feature, the iSBC 86/12A microcomputer has the ability to protect on-board memory from MULTIBUS access to any contiguous 8K-byte segments (or 16K-byte segments with iSBC 300 module). These features allow the multiprocessor systems to establish local memory for each processor and shared system (MULTIBUS) memory configurations where the total system memory size (including local on-board memory) can exceed 1 megabyte without addressing conflicts.

EPROM Capabilities

Four sockets are provided for up to 16K bytes of nonvolatile read only memory on the iSBC 86/12A board. EPROM may be added in 2K-byte increments up to a maximum of 4K bytes by using Intel® 2758 electrically programmable ROMs (EPROMs); in 4K-byte increments up to 8K bytes by using Intel 2716 EPROMs; or in 8K-byte increments up to 16K bytes using Inel 2732 EPROMs. On-board EPROM is accessed via 16-bit data paths. On-board EPROM capacity may be expanded to 32K bytes with the addition of the iSBC 340 16K-byte MULTIMODUE EPROM option. It provides an additional four sockets for Intel 2732 EPROMs. With user modification of the iSBC 86/12A's on-board memory and MULITBUS address decode, Intel 2758 and 2716 EPROMs may be optionally supported. System memory size is easily expanded by the addition of MULTIBUS system bus compatible memory boards available in the iSBC product family.

Parallel I/O Interface

The iSBC 86/12A single board computer contains 24 programmable parallel I/O lines implemented using the Intel 8255A Programmable Peripheral Interface. The system software is used to configure the I/O lines in any combination of unidirectional input/output and bidirectional ports indicated in Table 1. Therefore, the I/O interface may be customized to meet specific peripheral requirements. In order to take full advantage of the large number of possible I/O configurations, sockets are provided for interchangeable I/O line drivers and terminators. Hence, the flexibility of the I/O interface is further enhanced by the capability of selecting the appropriate combination of optional line drivers and terminators to provide the required sink current, polarity, and drive/termination characteristics for each application. The 24 programmable I/O lines and signal ground lines are brought out to a 50-pin edge connector that mates with flat, woven, or round cable.

Serial I/O

A programmable communications interface using the Intel 8251A Universal Synchronous/Asynchronous Receiver/Transmitter (USART) is contained on the iSBC 86/12A board. A software selectable baud rate generator provides the USART with all common communication

Table 1. Input/Output Port Modes of Operation

Port	Lines (qty)	Mode of Operation					
		Unidirectional				Bidirectional	Control
		Input		Output			
		Latched	Latched & Strobed	Latched	Latched & Strobed		
1	8	X	X	X	X	X	
2	8	X	X	X	X		
3	4	X		X			X[1]
	4	X		X			X[1]

Note
1. Part of port 3 must be used as a control port when either port 1 or port 2 are used as a latched and strobed input or a latched and strobed output port or port 1 is used as a bidirectional port.

frequencies. The USART can be programmed by the system software to select the desired asynchronous or synchronous serial data transmission technique (including IBM Bi-Sync). The mode of operation (i.e., synchronous or asynchronous), data format, control character format, parity, and baud rate are all under program control. The 8251A provides full duplex, double buffered transmit and receive capability. Parity, overrun, and framing error detection are all incorporated in the USART. The RS232C compatible interface on each board, in conjunction with the USART, provides a direct interface to RS232C compatible terminals, cassettes, and asynchronous and synchronous modems. The RS232C command lines, serial data lines, and signal ground line are brought out to a 26 pin edge connector that mates with RS232C compatible flat or round cable. The iSBC 530 Teletypewriter Adapter provides an optically isolated interface for those systems requiring a 20 mA current loop. The iSBC 530 unit may be used to interface the iSBC 86/12A board to teletypewriters or other 20 mA current loop equipment.

Programmable Timers

The iSBC 86/12A board provides three independent, fully programmable 16-bit interval timers/event counters utilizing the Intel 8253 Programmable Interval Timer. Each counter is capable of operating in either BCD or binary modes. Two of these timers/counters are available to the systems designer to generate accurate time intervals under software control. Routing for the outputs and gate/trigger inputs of two of these counters is jumper selectable. The outputs may be independently routed to the 8259A Programmable Interrupt Controller and to the I/O line drivers associated with the 8255A Programmable Peripheral Interface, or may be routed as inputs to the 8255A chip. The gate/trigger inputs may be routed to I/O terminators associated with the 8255A or as output connections from the 8255A. The third interval timer in the 8253 provides the programmable baud rate generator for the iSBC 86/12A board RS232C USART serial port. In utilizing the iSBC 86/12A board the systems designer simply configures, via software, each timer independently to meet system requirements. Whenever a given time delay or count is needed, software commands to the programmable timers/event counters select the desired function. Seven functions are available, as shown in Table 2. The contents of each counter may be read at any time during system operation with simple read operations for event counting applications, and special commands are included so that the contents can be read "on the fly".

MULTIBUS® System Bus and Multimaster Capabilities

The MULTIBUS system bus features asynchronous data transfers for the accommodation of devices with various transfer rates while maintaining maximum throughput. Twenty address lines and sixteen separate data lines eliminate the need for address/data multiplexing/demultiplexing logic used in other systems, and allow for data transfer rates up to 5 megawords/sec. A failsafe timer is included in the iSBC 86/12A board which can be used to generate an interrupt if an addressed device does not respond within 6 msec.

Table 2. Programmable Timer Functions

Function	Operation
Interrupt on terminal count	When terminal count is reached, an interrupt request is generated. This function is extremely useful for generation of real-time clocks.
Programmable one-shot	Output goes low upon receipt of an external trigger edge or software command and returns high when terminal count is reached. This function is retriggerable.
Rate generator	Divide by N counter. The output will go low for one input clock cycle, and the period from one low going pulse to the next is N times the input clock period.
Square-wave rate generator	Output will remain high until one-half the count has been completed, and go low for the other half of the count.
Software triggered strobe	Output remains high until software loads count (N). N counts after count is loaded, output goes low for one input clock period.
Hardware triggered strobe	Output goes low for one clock period N counts after rising edge counter trigger input. The counter is retriggerable.
Event counter	On a jumper selectable basis, the clock input becomes an input from the external system. CPU may read the number of events occurring after the counting "window" has been enabled or an interrupt may be generated after N events occur in the system.

Multimaster Capabilities — The iSBC 86/12A board is a full computer on a single board with resources capable of supporting a great variety of OEM system requirements. For those applications requiring additional processing capacity and the benefits of multiprocessing (i.e., several CPUs and/or controllers logically sharing system tasks through communication over the system bus), the iSBC 86/12A board provides full MULTIBUS arbitration control logic. This control logic allows up to three iSBC 86/12A boards or other bus masters, including iSBC 80 family MULTIBUS compatible 8-bit single board computers, to share the system bus in serial (daisy chain) priority fashion and up to 16 masters to share the MULTIBUS system bus with the addition of an external priority network. The MULTIBUS arbitration logic operates synchronously with a MULTIBUS clock (provided by the iSBC 86/12A board or optionally provided directly from the MULTIBUS) while data is transferred via a handshake between the master and slave modules. This allows different speed controllers to share resources on the same bus, and transfers via the bus proceed asynchronously. Thus, transfer speed is dependent on transmitting and

receiving devices only. This design prevents slow master modules from being handicapped in their attempts to gain control of the bus, but does not restrict the speed at which faster modules can transfer data via the same bus. The most obvious applications for the master-slave capabilities of the bus are multiprocessor configurations, high speed peripheral control, but are by no means limited to these three.

Interrupt Capablity

The iSBC 86/12A board provides 9 vectored interrupt levels. The highest level is the NMI (Non-maskable Interrupt) line which is directly tied to the 8086 CPU. This interrupt cannot be inhibited by software and is typically used for signalling catastrophic events (i.e., power failure). On servicing this interrupt, program control will be implicitly transferred through location 00008_H. The Intel 8259A Programmable Interrupt Controller (PIC) provides vectoring for the next eight interrupt levels. As shown in Table 3, a selection of four priority processing modes is available to the systems designer for use in designing request processing configurations to match system requirements. Operating mode and priority assignments may be reconfigured dynamically via software at any time during system operation. The PIC accepts interrupt requests from the programmable parallel and serial I/O interfaces, the programmable timers, the system bus, or directly from peripheral equipment. The PIC then determines which of the incoming requests is of the highest priority, determines whether this request is of higher priority than the level currently being serviced, and, if appropriate, issues an interrupt to the CPU. Any combination of interrupt levels may be masked, via software, by storing a single byte in the interrupt mask register of the PIC. The PIC generates a unique memory address for each interrupt level. These addresses are equally spaced at 4 byte intervals. This 32-byte block may begin at any 32-byte boundary in the lowest 1K-bytes of memory,* and contains unique instruction pointers and code segment offset values (for expanded memory operation) for each interrupt level. After acknowledging an interrupt and obtaining a device identifier byte from the 8259A PIC, the CPU will store its status flags on the stack and execute an indirect CALL instruction through the vector location (derived from the device identifier) to the interrupt service routine. In systems requiring additional interrupt levels, slave 8259A PIC's may be interfaced via the MULTIBUS system bus, to generate additional vector addresses, yielding a total of 65 unique interrupt levels.

Interrupt Request Generation — Interrupt requests may originate from 18 sources. Two jumper selectable interrupt requests can be automatically generated by the programmable peripheral interface when a byte of information is ready to be transferred to the CPU (i.e., input buffer is full) or a byte of information has been transferred to a peripheral device (i.e., output buffer is empty). Two jumper selectable interrupt requests can be automatically generated by the USART when a character is ready to be transferred to the CPU (i.e., receive channel buffer is full, or a character is ready to be transmitted (i.e., transmit channel data buffer is empty). A jumper selectable request can be generated by each of the programmable timers. An additional interrupt request line may be jumpered directly from the parallel I/O driver terminator section. Eight prioritized interrupt request lines allow the iSBC 86/12A board to recognize and service interrupts originating from peripheral boards interfaced via the MULTIBUS system bus. The MULTIBUS fail safe timer of the iSBC 337 processor and the exception and error output signal also can be selected as interrupt sources.

Table 3. Programmable Interrupt Modes

Mode	Operation
Fully nested	Interrupt request line priorities fixed at 0 as highest, 7 as lowest.
Auto-rotating	Equal priority. Each level, after receiving service, becomes the lowest priority level until next interrupt occurs.
Specific priority	System software assigns lowest priority level. Priority of all other levels based in sequence numerically on this assignment.
Polled	System software examines priority-encoded system interrupt status via interrupt status register.

Power-Fail Control

Control logic is also included to accept a power-fail interrupt in conjunction with the AC-low signal from the iSBC 635 and iSBC 640 Power Supply or equivalent.

Expansion Capabilities

Memory and I/O capacity may be expanded and additional functions added using Intel MULTIBUS compatible expansion boards. Memory may be expanded by adding user specified combinations of RAM boards, EPROM boards, or combination boards. Input/output capacity may be increased by adding digital I/O and analog I/O expansion boards. Mass storage capability may be achieved by adding single or double density diskette controllers, or hard disk controllers. Modular expandable backplanes and cardcages are available to support multiboard systems.

Note: Certain system restrictions may be incurred by the inclusion of some of the iSBC 80 family options in an iSBC 86/12A system. Consult the Intel OEM Microcomputer System Configuration Guide for specific data.

***Note:** The first 32 vector locations are reserved by Intel for dedicated vectors. Users who wish to maintain compatibility with present and future Intel products should not use these locations for user-defined vector addresses.

System Development Capabilities

The development cycle of iSBC 86/12A products can be significantly reduced by using the Intellec® series microcomputer development system. The Assembler, High Level Languages, Locating Linker, Library Manager, Text Editor and System Monitor are all supported by the ISIS-II disk-based operating system.

In-Circuit Emulator — ICE™-86 in-circuit emulator provides the necessary link between the software development environment provided by the Intellec system and the "target" iSBC 86/12A execution system. In addition to providing the mechanism for loading executable code and data into the iSBC 86/12A board, ICE-86 in-circuit emulator provides a sophisticated command set to assist in debugging software and final integration of the user hardware and software. ICE-86 in-circuit emulator maximizes the use of available development resources by allowing Intellec resident resources (e.g., memory and peripherals) to be accessed by software running on the target iSBC 86/12A system. In addition, software can be executed without an iSBC 86/12A execution vehicle, in 2K bytes of RAM resident in the ICE-86 system itself. Symbolic references to instruction and data locations can be made through ICE-86 in-circuit emulator to allow the user to reference memory locations with assigned names.

PL/M-86 — Intel's high level programming language, PL/M-86, is also available as an Intellec Microcomputer Development System option. PL/M-86 provides the capability to program in a natural, algorithmic language and eliminates the need to manage register usage or allocate memory. PL/M-86 programs can be written in a much shorter time than assembly language programs for a given application. PL/M-86 includes byte and word, integer, pointer and floating point (32-bit) data types and also includes conditional compilation and macro features.

SPECIFICATIONS

Word Size

Instruction — 8, 16, 24, or 32 bits
Data — 8, 16 bits

Cycle Time

Basic Instruction Cycle — 1.2 μsec
— 400 nsec (assumes instruction in the queue)

Note:
Basic instruction cycle is defined as the fastest instruction time (i.e., two clock cycles)

Memory Capacity

On-Board Read Only Memory — 16K bytes (sockets only); expandable to 32K bytes with iSBC 340 16K-byte MULTIMODULE EPROM option.

On-Board RAM — 32K bytes; expandable to 64K bytes with iSBC 300 32K-byte MULTIMODULE RAM option.

Off-Board Expandion — Up to 1 megabyte in user specified combinations of RAM and EPROM.

Note:
Read only memory may be added in 2K, 4K, or 8K-byte increments.

Memory Addressing

On-Board EPROM — $FF000\text{-}FFFFF_H$ (using 2758 EPROMs); $FE000\text{-}FFFFF_H$ (using 2716 EPROMs); $FC000\text{-}FFFFF_H$ (using 2732 EPROMS); $F8000\text{-}FFFFF_H$ (with iSBC 340 EPROM option and four additional 2732 EPROMs).

On-Board RAM — 32K bytes of dual port RAM. Optionally expandable to 64K bytes with iSBC 300 RAM option.

CPU Access — 32K bytes: $00000\text{-}07FFF_H$; 64K bytes: $00000\text{-}0FFFF_H$.

MULTIBUS Access — Jumper selectable for any 8K-byte boundary, but not crossing a 128K-byte boundary. Access for 8K, 16K, 24K or 32K (16K, 32K, 48K, 64K with iSBC 300 option) bytes may be selected for on-board CPU use only.

I/O Capacity

Parallel — 24 programmable lines using one 8255A.
Serial — 1 programmable line using one 8251A.

I/O Addressing

On-Board Programmable I/O

Port	8255A				USART	
	1	2	3	Control	Data	Control
Address	C8	CA	CC	CE	D8 or DC	DA or DE

Serial Communications Characteristics

Synchronous — 5—8 bit characters; internal or external character synchronization; automatic sync insertion.

Asynchronous — 5—8 bit characters; break character generation; 1, 1½, or 2 stop bits; false start bit detection.

Baud Rates

Frequency (kHz) (Software Selectable)	Baud Rate (Hz)		
	Synchronous	Asynchronous	
		÷ 16	÷ 64
153.6	—	9600	2400
76.8	—	4800	1200
38.4	38400	2400	600
19.2	19200	1200	300
9.6	9600	600	150
4.8	4800	300	75
2.4	2400	150	—
1.76	1760	110	—

Note:
Frequency selected by I/O write of appropriate 16-bit frequency factor to baud rate register (8253 Timer 2).

Interrupts

Addresses for 8259A Registers (Hex notation I/O address space)

C0 or C4 Write: Initialization Command Word 1 (ICW1) and Operation Control Words 2 and 3 (OCW2 and OCW3)

Read: Status and Poll Registers

C2 or C6 Write: ICW2, ICW3, ICW4, 0CW1 (Mask Register)

Read: 0CW1 (Mask Register)

Note:
Several registers have the same physical address; sequence of access and one data bit of control word determine which register will respond.

Interrupt Levels — 8086 CPU includes a non-maskable Interrupt (NMI) and a maskable interrupt (INTR). NMI interrupt is provided for catastrophic events such as power failure. NMI vector address is 00008. INTR interrupt is driven by on-board 8259A PIC, which provides 8-bit identifier of interrupting device to CPU. CPU multiplies identifier by four to derive vector address. Jumpers select interrupts from 18 sources without necessity of external hardware. PIC May be programmed to accommodate edge-sensitive or level-sensitive inputs.

Timers

Register Addresses (Hex notation, I/O address space)

D0 Timer 0
D2 Timer 1
D4 Timer 2
D6 Control register

Note:
Timer counts are loaded as two sequential output operations to same address as given.

Input Frequencies

Reference: 2.46 MHz ±0.1% (0.041 μs period, nominal); 1.23 MHz ±0.1% (0.81 μs period,. nominal); or 153.60 kHz ±0.1% (6.51 μs period nominal).

Note:
Above frequencies are user selectable.

Event Rate: 2.46 MHz max

Output Frequencies/Timing Intervals

Function	Single Timer/Counter		Dual Timer/Counter (Two Timers Cascaded)	
	Min	Max	Min	Max
Real-time interrupt	1.63 μs	427.1 ms	3.26 s	466.50 min
Programmable one-shot	1.63 μs	427.1 ms	3.26 s	466.50 min
Rate generator	2.342 Hz	613.5 kHz	0.000036 Hz	306.8 kHz
Square-wave rate generator	2.342 Hz	613.5 kHz	0.000036 Hz	306.8 kHz
Software triggered strobe	1.63 μs	427.1 ms	3.26 s	466.50 min
Hardware triggered strobe	1.63 μs	427.1 ms	3.26 s	466.50 min
Event counter	—	2.46 MHz	—	—

Interfaces

MULTIBUS — All signals TTL compatible
Parallel I/O — All signals TTL compatible
Interrupt Requests — All TTL compatible
Timer — All signals TTL compatible
Serial I/O — RS232C compatible, data set configuration

System Clock (8086 CPU)

5.00 MHz ± 0.1%

Auxiliary Power

An auxiliary power bus is provided to allow separate power to RAM for systems requiring battery backup of read/write memory. Selection of this auxiliary RAM power bus is made via jumpers on the board.

Connectors

Interface	Pins (qty)	Centers (in.)	Mating Connectors
Bus	86	0.156	VIKING 3KH43/9AMK12
Parallel I/O	50	0.1	3M 3415-000
Serial I/O	26	0.1	3M 3462-000

Memory Protect

An active low TTL compatible memory protect signal is brought out on the auxiliary connector which, when asserted, disables read/write access to RAM memory on the board. This input is provided for the protection of RAM contents during system power down sequences.

Line Drivers and Terminators

I/O Drivers — The following line drivers are all compatible with the I/O driver sockets on the iSBC 86/12A board

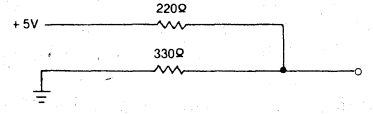
Driver	Characteristic	Sink Current (mA)
7438	I,OC	48
7437	I	48
7432	NI	16
7426	I,OC	16
7409	NI,OC	16
7408	NI	16
7403	I,OC	16
7400	I	16

Note:
I = inverting; NI = non-inverting; OC = open collector.

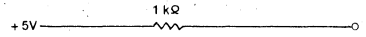
Port 1 of the 8255A has 20 mA totem-pole bidirectional drivers and 1 kΩ terminators.

I/O Terminators — 220Ω/330Ω divider or 1 kΩ pullup

220Ω/330Ω (ISBC 901 OPTION)



1 K Ω (ISBC 902 OPTION)



Bus Drivers

Function	Characteristic	Sink Current (mA)
Data	Tri-state	50
Address	Tri-state	50
Commands	Tri-state	32

Physical Characteristics

Width — 12.00 in. (30.48 cm)
Height — 6.75 in. (17.15 cm)
Depth — 0.70 in. (1.78 cm)
Weight — 19 oz. (539 gm)

Electrical Characteristics

DC Power Requirements

Configuration	Current Requirements			
	V_{CC} = +5V ±5% (max)	V_{DD} = +12V ±5% (max)	V_{BB} = −5V ±5% (max)	V_{AA} = −12V ±5% (max)
Without EPROM[1]	5.2A	350 mA	—	40 mA
RAM Only[3]	390 mA	40 mA	1.0 mA	—
With iSBC 530[4]	5.2A	450 mA	—	140 mA
With 4K EPROM[5] (using 2758)	5.5A	350 mA	—	40 mA
With 8K EPROM[5] (using 2716)	5.5A	350 mA	—	40 mA
With 16K EPROM[5] (using 2732)	5.4A	350 mA	—	40 mA

Notes:

1. Does not include power for optional EPROM, I/O drivers, and I/O terminators.
2. Does not include power required for optional EPROM, I/O drivers, and I/O terminators.
3. RAM chips powered via auxiliary power bus.
4. Does not include power for optional EPROM, I/O drivers, and I/O terminators. Power for iSBC 530 is supplied via serial port connector.
5. Includes power required for four EPROM chips, and I/O terminators installed for 16 I/O lines; all terminator inputs low.

Environmental Characteristics

Operating Temperature — 0°C to 55°C
Relative Humidity — to 90% (without condensation)

Reference Manual

9803074-01 — iSBC 86/12A Single Board Computer Hardware Reference Manual (NOT SUPPLIED)

Reference manuals are shipped with each product only if designated SUPPLIED (see above). Manuals may be ordered from any Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

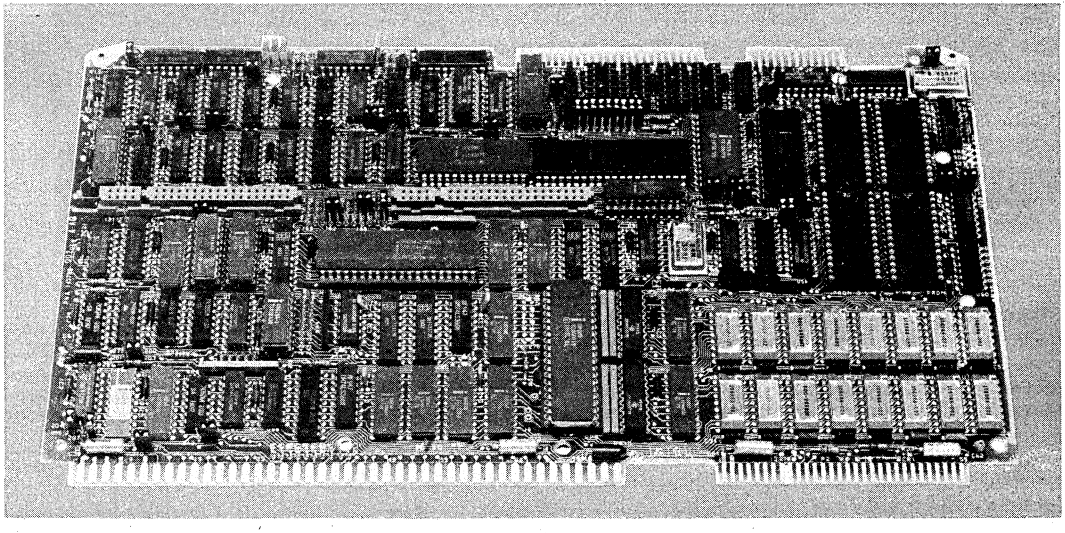
ORDERING INFORMATION

Part Number	Description
SBC 86/12A	Single Board Computer with 32K bytes RAM

iSBC® 86/14 and iSBC® 86/30 SINGLE BOARD COMPUTERS

- iAPX 86/10 (8086-2) Microprocessor with 5 or 8 MHz CPU clock
- Fully software compatible with iSBC® 86/12A Single Board Computer
- Optional iAPX 86/20 Numeric Data Processor with iSBC® 337 MULTIMODULE™ processor
- 32K/128K bytes of dual-port read/write memory expandable on-board to 256K bytes with on-board refresh
- Sockets for up to 64K bytes of JEDEC 24/28-pin standard memory devices
- Two iSBX™ bus connectors
- 24 programmable parallel I/O lines
- Programmable synchronous/asynchronous RS232C compatible serial interface with software selectable baud rates
- Two programmable 16-bit BCD or binary timers/event counters
- 9 Levels of vectored interrupt control, expandable to 65 levels
- MULTIBUS® interface for multimaster configurations and system expansion
- Supported by a complete family of single board computers, memory, digital and analog I/O, peripheral controllers, packaging and software

The iSBC 86/14 and iSBC 86/30 Single Board Computers are members of Intel's complete line of OEM microcomputer systems which take full advantage of Intel's technology to provide economical, self-contained, computer-based solutions for OEM applications. Each board is a complete computer system on a single 6.75 x 12.00-in. printed circuit card distinguished by RAM memory content with 32K bytes and 128K bytes provided on the iSBC 86/14 and iSBC 86/30 board, respectively. The CPU, system clock, read/write memory, nonvolatile read only memory, I/O ports and drivers, serial communications interface, priority interrupt logic and programmable timers, all reside on the boards.



April 1984
Order Number: 280007-001

FUNCTIONAL DESCRIPTION

Central Processing Unit

The central processor for the iSBC 86/XX[1] boards is Intel's iAPX 86/10 (8086-2) CPU. A clock rate of 8 MHz is supported with a jumper selectable option of 5 MHz. The CPU architecture includes four 16-bit byte addressable data registers, two 16-bit memory base pointer registers and two 16-bit index registers, all accessed by a total of 24 operand addressing modes for comprehensive memory addressing and for support of the data structures required for today's structured, high level languages as well as assembly language.

[1] iSBC 86/XX designates both the iSBC 86/14 and iSBC 86/30 CPU boards.

Instruction Set

The 8086 instruction repertoire includes variable length instruction format (including double operand instructions), 8-bit and 16-bit signed and unsigned arithmetic operators for binary, BCD and unpacked ASCII data, and iterative word and byte string manipulation functions.

For enhanced numerics processing capability, the iSBC 337 MULTIMODULE Numeric Data Processor extends the iAPX 86/10 architecture and data set. Over 60 numeric instructions offer arithmetic, trigonometric, transcendental, logarithmic and exponential instructions. Supported data types include 16, 32, and 64-bit integer, and 32 and 64-bit floating point, 18-digit packed BCD and 80-bit temporary.

Architectural Features

A 6-byte instruction queue provides pre-fetching of sequential instructions and can reduce the 750 nsec minimum instruction cycle to 250 nsec for queued instructions. The stack-oriented architecture readily supports modular programming by facilitating fast, simple, inter-module communication, and other programming constructs needed for asynchronous real-time systems. The memory expansion capabilities offer a 1 megabyte addressing range. The dynamic relocation scheme allows ease in segmentation of pure procedure and data for efficient memory utilization. Four segment registers (code, stack, data, extra) contain program loaded offset values which are used to map 16-bit addresses to 20-bit addresses. Each register maps 64K bytes at a time and activation of a specific register is controlled explicitly by program control and is also selected implicitly by specific functions and instructions.

RAM Capabilities

The iSBC 86/14 and iSBC 86/30 microcomputers contain 32K bytes and 128K bytes of dual-port dynamic RAM, respectively. In addition, on-board

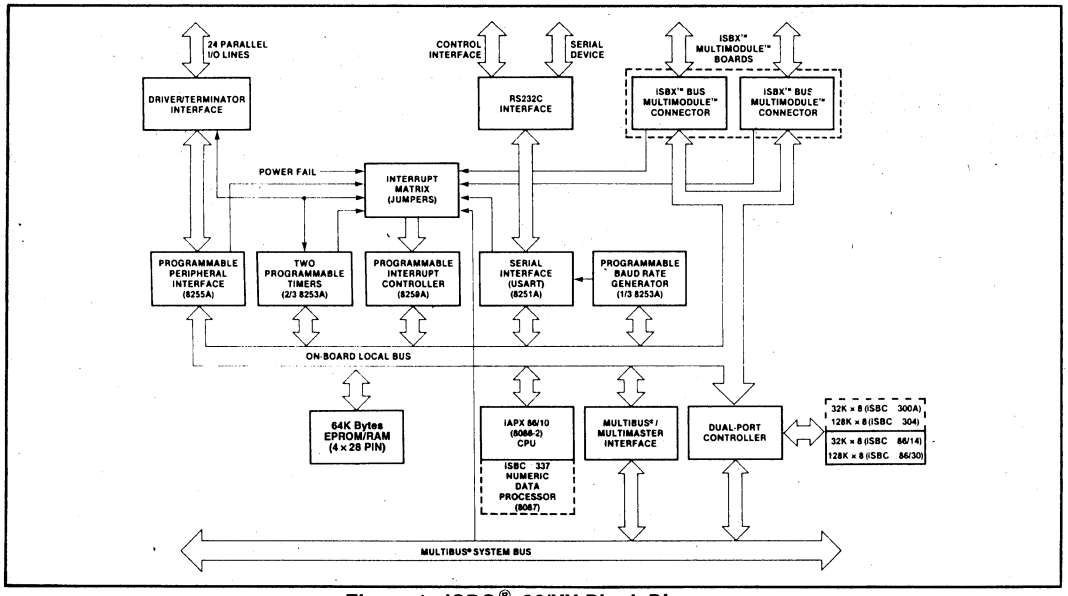


Figure 1. iSBC® 86/XX Block Diagram

RAM may be doubled on each microcomputer by optionally adding RAM MULTIMODULE boards. The on-board RAM may be expanded to 256K bytes with the iSBC 304 MULTIMODULE Board mounted onto the iSBC 86/30 board. Likewise, the iSBC 86/14 microcomputer may be expanded to 64K bytes with the iSBC 300A MULTIMODULE option. The dual-port controller allows access to the on-board RAM (including RAM MULTIMODULE options) from the iSBC 86/XX boards and from any other MULTIBUS master via the system bus. Segments of on-board RAM may be configured as a private resource, protected from MULTIBUS system access. The amount of memory allocated as a private resource may be configured in increments of 25% of the total on-board memory ranging from 0% to 100% (optional RAM MULTIMODULE boards double the increment size). These features allow the multiprocessor systems to establish local memory for each processor and shared system memory configurations where the total system memory size (including local on-board memory) can exceed one megabyte without addressing conflicts.

EPROM Capabilities

Four 28-pin sockets are provided for the use of Intel 2716s, 2732As, 2764s, 27128s, and their respective ROMs. When using 27128s, the on-board EPROM capacity is 64K bytes. Other JEDEC standard pinout devices are also supported, including byte-wide static RAMs.

Parallel I/O Interface

The iSBC 86/XX Single Board Computers contain 24 programmable parallel I/O lines implemented using the Intel 8255A Programmable Peripheral Interface. The system software is used to configure the I/O lines in any combination of unidirectional input/output and bidirectional ports indicated in Table 1. In order to take advantage of the large number of possible I/O configurations, sockets are provided for interchangeable I/O line drivers and terminators, allowing the selection of the appropriate combination of optional line drivers and terminators with the required drive/termination characteristics. The 24 programmable I/O lines and signal ground lines are brought out to a 50-pin edge connector.

Serial I/O

A programmable communications interface using the Intel 8251A Universal Synchronous/Asynchronous Receiver/Transmitter (USART) is contained on the iSBC 86/XX boards. A software selectable baud rate generator provides the USART with all common communication frequencies. The mode of operation (i.e., synchronous or asynchronous), data format, control character format, parity, and baud rate are all under program control. The 8251A provides full duplex, double buffered transmit and receive capability. Parity, overrun, and framing error detection are all incorporated in the USART. The RS232C command lines, serial data lines and signal ground line are brought out to a 26-pin edge connector.

Programmable Timers

The iSBC 86/XX boards provide three independent, fully programmable 16-bit interval timers/event counters utilizing the Intel 8253 Programmable Interval Timer. Each counter is capable of operating in either BCD or binary modes. Two of these timers/counters are available to the systems designer to generate accurate time intervals under software control. Routing for the outputs and gate/trigger inputs of two of these counters is jumper selectable. The outputs may be indepen-

Table 1. Input/Output Port Modes of Operation

Port	Lines (qty)	Mode of Operation					
		Unidirectional				Bidirectional	Control
		Input		Output			
		Latched	Latched & Strobed	Latched	Latched & Strobed		
1	8	X	X	X	X	X	
2	8	X	X	X	X		
3	4	X		X			X[1]
	4	X		X			X[1]

NOTE:
1. Part of port 3 must be used as a control port when either port 1 or port 2 are used as a latched and strobed input or a latched and strobed output port or port 1 is used as a bidirectional port.

dently routed to the 8259A Programmable Interrupt Controller and to the I/O terminators associated with the 8255A to allow external devices or an 8255A port to gate the timer or to count external events. The third interval timer in the 8253 provides the programmable baud rate generator for the iSBC 86/XX boards' RS232C USART serial port. The system software configures each timer independently to select the desired function. Seven functions are available as shown in Table 2. The contents of each counter may be read at any time during system operation.

Table 2. Programmable Timer Functions

Function	Operation
Interrupt on terminal count	When terminal count is reached, an interrupt request is generated. This function is extremely useful for generation of real-time clocks.
Programmable one-shot	Output goes low upon receipt of an external trigger edge or software command and returns high when terminal count is reached. This function is retriggerable.
Rate generator	Divide by N counter. The output will go low for one input clock cycle, and the period from one low going pulse to the next is N times the input clock period.
Square-wave rate generator	Output will remain high until one-half the count has been completed, and go low for the other half of the count.
Software triggered strobe	Output remains high until software loads count (N). N counts after count is loaded, output goes low for one input clock period.
Hardware triggered strobe	Output goes low for one clock period N counts after rising edge counter trigger input. The counter is retriggerable.
Event counter	On a jumper selectable basis, the clock input becomes an input from the external system. CPU may read the number of events occurring after the counter "window" has been enabled or an interrupt may be generated after N events occur in the system.

iSBX™ MULTIMODULE™ On-Board Expansion

Two 8/16-bit iSBX MULTIMODULE connectors are provided on the iSBC 86/XX microcomputers. Through these connectors, additional on-board I/O functions may be added. iSBX MULTIMODULE boards optimally support functions provided by VLSI peripheral components such as additional parallel and serial I/O, analog I/O, small mass storage device controllers (e.g., cassettes and floppy disks), and other custom interfaces to meet specific needs. By mounting directly on the single board computer, less interface logic, less power, simpler packaging, higher performance, and lower cost result when compared to other alternatives such as MULTIBUS form factor compatible boards. The iSBX connectors on the iSBC 86/XX boards provide all signals necessary to interface to the local on-board bus, including 16 data lines for maximum data transfer rates. iSBX MULTIMODULE boards designed with 8-bit data paths and using the 8-bit iSBX connector are also supported on the iSBC 86/XX microcomputers. A broad range of iSBX MULTIMODULE options are available in this family from Intel. Custom iSBX modules may also be designed for use on the iSBC 86/XX boards. An iSBX bus interface specification and iSBX connectors are available from Intel.

MULTIBUS® SYSTEM BUS AND MULTIMASTER CAPABILITIES

Overview

The MULTIBUS system bus is Intel's industry standard microcomputer bus structure. Both 8 and 16-bit single board computers are supported on the MULTIBUS structure with 24 address and 16 data lines. In its simplest application, the MULTIBUS system bus allows expansion of functions already contained on a single board computer (e.g., memory and digital I/O). However, the MULTIBUS structure also allows very powerful distributed processing configurations with multiple processors and intelligent slave I/O, and peripheral boards capable of solving the most demanding microcomputer applications. The MULTIBUS system bus is supported with a broad array of board level products, LSI interface components, detailed published specifications and application notes.

Expansion Capabilities

Memory and I/O capacity may be expanded and additional functions added using Intel MULTIBUS compatible expansion boards. Memory may be expanded by adding user specified combinations of RAM boards, EPROM boards, or combination boards. Input/output capacity may be added with digital I/O and analog I/O expansion boards. Mass storage capability may be achieved by adding single or double density diskette controllers, or

hard disk controllers. Modular expandable backplanes and cardcages are available to support multiboard systems.

Multimaster Capabilities

For those applications requiring additional processing capacity and the benefits of multiprocessing (i.e., several CPUs and/or controllers logically sharing system tasks through communication of the system bus), the iSBC 86/XX boards provide full MULTIBUS arbitration control logic. This control logic allows up to three iSBC 86/XX boards or other bus masters, including iSBC 80 family MULTIBUS compatible 8-bit single board computers to share the system bus using a serial (daisy chain) priority scheme and allows up to 16 masters to share the MULTIBUS system bus with an external parallel priority decoder. In addition to the multiprocessing configurations made possible with multimaster capability, it also provides a very efficient mechanism for all forms of DMA (Direct Memory Access) transfers.

Interrupt Capability

The iSBC 86/XX boards provide 9 vectored interrupt levels. The highest level is the NMI (Non-Maskable Interrupt) line which is directly tied to the 8086 CPU. This interrupt is typically used for signaling catastrophic events (e.g., power failure). The Intel 8259A Programmable Interrupt Controller (PIC) provides control and vectoring for the next eight interrupt levels. As shown in Table 3, a selection of four priority processing modes is available for use in designing request processing configurations to match system requirements for efficient interrupt servicing with minimal latencies. Operating mode and priority assignments may be reconfigured dynamically via software at any time during system operation. The PIC accepts interrupt requests from all on-board I/O resources and from the MULTIBUS system bus. The PIC then resolves requests according to the selected mode and, if appropriate, issues an interrupt to the CPU. Any combination of interrupt levels may be masked via software, by storing a single byte in the interrupt mask register of the PIC. In systems requiring additional interrupt levels, slave 8259A PICs may be interfaced via the MULTIBUS system bus, to generate additional vector addresses, yielding a total of 65 unique interrupt levels.

Table 3. Programmable Interrupt Modes

Mode	Operation
Fully nested	Interrupt request line priorities fixed at 0 as highest, 7 as lowest.
Auto-rotating	Equal priority. Each level, after receiving service, becomes the lowest priority level until next interrupt occurs.
Specific priority	System software assigns lowest priority level. Priority of all other levels based in sequence numerically on this assignment.
Polled	System software examines priority-encoded system interrupt status via interrupt status register.

Interrupt Request Generation

Interrupt requests to be serviced by the iSBC 86/XX boards may originate from 28 sources. Table 4 includes a list of devices and functions supported by interrupts. All interrupt signals are brought to the interrupt jumper matrix where any combination of interrupt sources may be strapped to the desired interrupt request level on the 8259A PIC or the NMI input to the CPU directly.

Power-Fail Control and Auxiliary Power

Control logic is also included to accept a power-fail interrupt in conjunction with the AC-low signal from the iSBC 635 and iSBC 640 Power Supply or equivalent, to initiate an orderly shut down of the system in the event of a power failure. Additionally, an active-low TTL compatible memory protect signal is brought out on the auxiliary connector which, when asserted, disables read/write access to RAM memory on the board. This input is provided for the protection of RAM contents during system power-down sequences. An auxiliary power bus is also provided to allow separate power to RAM for systems requiring battery back-up of read/write memory. Selection of this auxiliary RAM power bus is made via jumpers on the board.

System Development Capabilities

The development cycle of iSBC 86/XX products can be significantly reduced and simplified by using either the System 86/330 or the Intellec Series Microcomputer Development Systems. The Assembler, Locating Linker, Library Manager, Text Editor and System Monitor are all supported by the ISIS-II disk-based operating system. To facilitate conversion of 8080A/8085A assembly language programs to run on the iSBC 86/XX boards, CONV-86 is available under the ISIS-II operating system.

IN-CIRCUIT EMULATOR

The Intellec ICE-86 In-Circuit Emulator provides the necessary link between the software development environment provided by the Intellec system and the "target" iSBC 86/XX execution system. In addition to providing the mechanism for loading executable code and data into the iSBC 86/XX boards, the ICE-86 In-Circuit Emulator provides a sophisticated command set to assist in debugging software and final integration of the user hardware and software.

PL/M-86

Intel's system's implementation language, PL/M-86, is standard in the System 86/330 and is also available as an Intellec Microcomputer Development System option. PL/M-86 provides the capability to program in algorithmic language and eliminates the need to manage register usage or allocate memory while still allowing explicit control of the system's resources when needed. FORTRAN 86 and PASCAL 86 are also available on Intellec or 86/330 systems.

Run-Time Support

Intel also offers two run-time support packages; iRMX 88 Realtime Multitasking Executive and the iRMX 86 Operating System. The iRMX 88 executive is a simple, highly configurable and efficient foundation for small, high performance applications. Its multitasking structure establishes a solid foundation for modular system design and provides task scheduling and management, intertask communication and synchronization, and interrupt servicing for a variety of peripheral devices. Other configurable options include terminal handlers, disk file system, debuggers and other utilities. The iRMX 86 Operating System is a high functional operating system with a very rich set of features and options based on an object-oriented architecture. In addition to being modular and configurable, functions beyond the nucleus include a sophisticated file management and I/O system, and powerful human interface. Both packages are easily customized and extended by the user to match unique requirements.

Table 4. Interrupt Request Sources

Device	Function	Number of Interrupts
MULTIBUS® interface	Requests from MULTIBUS® resident peripherals or other CPU boards	8; may be expanded to 64 with slave 8259A PICs on MULTIBUS® boards
8255A Programmable Peripheral Interface	Signals input buffer full or output buffer empty; also BUS INTR OUT general purpose interrupt from driver/terminator sockets	3
8251A USART	Transmit buffer empty and receive buffer full	2
8253 Timers	Timer 0, 1 outputs; function determined by timer mode	2
iSBX™ connectors	Function determined by iSBX™ MULTIMODULE™ board	4 (2 per iSBX™ connector)
Bus fail safe timer	Indicates addressed MULTIBUS® resident device has not responded to command within 6 msec	1
Power fail interrupt	Indicates AC power is not within tolerance	1
Power line clock	Source of 120 Hz signal from power supply	1
External interrupt	General purpose interrupt from auxiliary (P2) connector on backplane	1
iSBC™ 337 MULTIMODULE™ Numeric Data Processor	Indicates error or exception condition	1
Parity error	Indicates on-board RAM parity error from iSBC™ 303 parity MULTIMODULE™ board (iSBC™ 86/14 option)	1
Edge-level conversion	Converts edge triggered interrupt request to level interrupt	1
OR-gate matrix	Outputs of OR-gates on-board for multiple interrupts	2

SPECIFICATIONS

Word Size

INSTRUCTION — 8, 16, 24, or 32 bits
DATA — 8, 16 bits

System Clock

5.00 MHz or 8.00 MHz ±0.1% (jumper selectable)

Cycle Time

BASIC INSTRUCTION CYCLE

8 MHz — 750 ns
— 250 ns (assumes instruction in the queue)
5 MHz — 1.2 μsec
— 400 ns (assumes instruction in the queue)

NOTE: Basic instruction cycle is defined as the fastest instruction time (i.e., two clock cycles).

Memory Cycle Time

RAM — 750 ns
EPROM — Jumper selectable from 500 ns to 875 ns

Memory Capacity/Addressing

ON-BOARD EPROM

Device	Total Capacity	Address Range
2716	8K bytes	FE000-FFFFF$_H$
2732A	16K bytes	FC000-FFFFF$_H$
2764	32K bytes	F8000-FFFFF$_H$
27128	64K bytes	F0000-FFFFF$_H$

NOTE: iSBC 86/XX EPROM sockets support JEDEC 24/28-pin standard EPROMs and RAMs.

ON-BOARD RAM

Board	Total Capacity	Address Range
iSBC 86/14	32K bytes	0-07FFF$_H$
iSBC 86/30	128K bytes	0-1FFFF$_H$

WITH MULTIMODULE™ RAM

Board	Total Capacity	Address Range
iSBC 300A (with iSBC 86/14)	64K bytes	0-0FFFF$_H$
iSBC 304 (with iSBC 86/30)	256K bytes	0-3FFFF$_H$

I/O Capacity

PARALLEL — 24 programmable lines using one 8255A.
SERIAL — 1 programmable line using one 8251A
iSBX™ MULTIMODULE™ — 2 iSBX boards

Serial Communications Characteristics

SYNCHRONOUS — 5-8 bit characters; internal or external character synchronization; automatic sync insertion

ASYNCHRONOUS — 5-8 bit characters; break character generation; 1, 1½, or 2 stop bits; false start bit detection

BAUD RATES

Frequency (kHz) (Software Selectable)	Baud Rate (Hz)		
	Synchronous	Asynchronous	
		÷16	÷64
153.6	—	9600	2400
76.8	—	4800	1200
38.4	38400	2400	600
19.2	19200	1200	300
9.6	9600	600	150
4.8	4800	300	75
2.4	2400	150	—
1.76	1760	110	—

NOTE: Frequency selected by I/O write of appropriate 16-bit frequency factor to baud rate register (8253 Timer 2).

Timers

INPUT FREQUENCIES

Reference: 2.46 MHz ±0.1% (0.041 μsec period, nominal); or 153.60 kHz ±0.1% (6.51 μsec period, nominal)

NOTE: Above frequencies are user selectable.

Event Rate: 2.46 MHz max

OUTPUT FREQUENCIES/TIMING INTERVALS

Function	Single Timer/Counter		Dual Timer/Counter (Cascaded)	
	Min	Max	Min	Max
Real-time Interrupt	1.63 μs	427.1 ms	3.26s	466.50 min
Programmable one-shot	1.63 μs	427.1 ms	3.26s	466.50 min
Rate generator	2.342 Hz	613.5 kHz	0.000036 Hz	306.8 kHz
Square-wave rate generator	2.342 Hz	613.5 kHz	0.000036 Hz	306.8 kHz
Software triggered strobe	1.63 μs	427.1 ms	3.26s	466.50 min
Hardware triggered strobe	1.63 μs	427.1 ms	3.26s	466.50 min
Event counter	—	2.46 MHz	—	—

Interfaces

MULTIBUS® — All signals TTL compatible
iSBX™ BUS — All signals TTL compatible
PARALLEL I/O — All signals TTL compatible
SERIAL I/O — RS232C compatible, configurable as a data set or data terminal

TIMER — All signals TTL compatible
INTERRUPT REQUESTS — All TTL compatible

Connectors

Interface	Double-Sided Pins	Centers (in.)	Mating Connectors
MULTIBUS® System	86	0.156	Viking 3KH43/9AMK12 Wire Wrap
iSBX™ Bus 8-Bit Data	36	0.1	iSBX 960-5
Parallel I/O (2)	50	0.1	3M 3415-000 Flat or TI H312125 Pins
Serial I/O	26	0.1	3M 3462-0001 Flat or AMP 88106-1 Flat

Line Drivers and Terminators

I/O DRIVERS — The following line drivers are all compatible with the I/O driver sockets on the iSBC 86/05 board

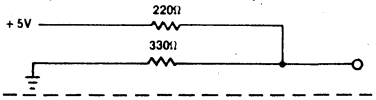
Driver	Characteristic	Sink Current (mA)
7438	I,OC	48
7437	I	48
7432	NI	16
7426	I,OC	16
7409	NI,OC	16
7408	NI	16
7403	I,OC	16
7400	I	16

NOTE: I = inverting; NI = non-inverting; OC = open collector.

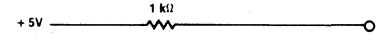
Port 1 of the 8255A has 20 mA totem-pole bidirectional drivers and 1 kΩ terminators

I/O TERMINATORS — 220Ω/330Ω divider or 1 kΩ pullup

220Ω/330Ω (iSBC™ 901 OPTION)



1 kΩ (iSBC™ 902 OPTION)



MULTIBUS® Drivers

Function	Characteristic	Sink Current (mA)
Data	Tri-State	32
Address	Tri-State	32
Commands	Tri-State	32
Bus Control	Open Collector	20

Physical Characteristics

WIDTH — 12.00 in. (30.48 cm)
HEIGHT — 6.75 in. (17.15 cm)
DEPTH — 0.70 in. (1.78 cm)
WEIGHT — 14 oz (388 gm)

Environmental Characteristics

OPERATING TEMPERATURE — 0°C to 55°C
RELATIVE HUMIDITY — to 90% (without condensation)

Electrical Characteristics

DC POWER REQUIREMENTS

Configuration	Current Requirements (All Voltages ±5%)		
	+5V	+12V	−12V
Without EPROM[1]	5.1A	25 mA	23 mA
RAM only[2]	600 mA	—	—
With 8K EPROM[3] (using 2716)	5.4A	25 mA	23 mA
With 16K EPROM[3] (using 2732A)	5.5A	25 mA	23 mA
With 32K EPROM[3] (using 2764)	5.6A	25 mA	23 mA

NOTES:
1. Does not include power for optional ROM/EPROM, I/O drivers, and I/O terminators.
2. RAM chips powered via auxiliary power bus in power-down mode.
3. Includes power required for 4 ROM/EPROM chips, and I/O terminators installed for 16 I/O lines; all terminator inputs low.

Environmental Characteristics

OPERATING TEMPERATURE — 0°C to 55°C
RELATIVE HUMIDITY — to 90% (without condensation)

Reference Manual

144044-001 — iSBC 86/14 and iSBC 86/30 Hardware Reference Manual (NOT SUPPLIED)

Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

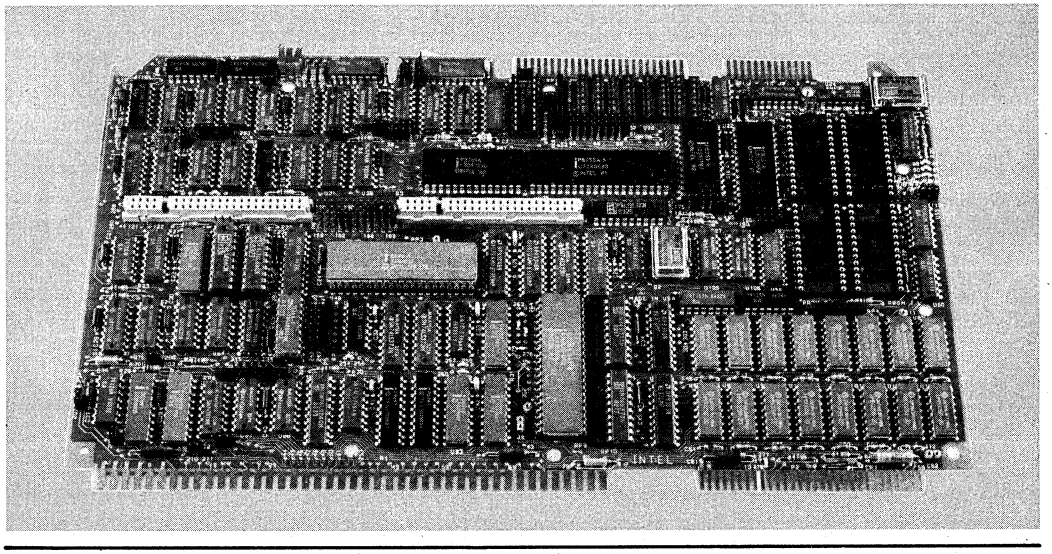
ORDERING INFORMATION

Part Number	Description
SBC 86/14	Single Board Computer
SBC 86/30	Single Board Computer

iSBC® 86/35
SINGLE BOARD COMPUTER

- iAPX 86/10 (8086-2) Microprocessor with 5 or 8 MHz CPU clock
- Optional iAPX 86/20 Numeric Data Processor with iSBC® 337 MULTIMODULE™ processor
- Upward compatible with iSBC 86/30 Single Board Computer
- 512K bytes of dual-port read/write memory expandable on-board to 640K or 1M bytes
- Sockets for up to 128K bytes of JEDEC 24/28-pin standard memory devices
- Two iSBX™ bus connectors
- 24 programmable parallel I/O lines
- Programmable synchronous/asynchronous RS232C compatible serial interface with software selectable baud rates
- Three programmable 16-bit BCD or binary timers/event counters
- 9 levels of vectored interrupt control, expandable off board to 65 levels
- MULTIBUS® interface for multimaster configurations and system expansion
- Supported by a complete family of single board computers, memory, digital and analog I/O, peripheral controllers, packaging and software

The iSBC 86/35 Single Board Computer is a member of Intel's complete line of OEM microcomputer systems that take full advantage of Intel's technology to provide economical, self-contained, computer-based solutions for OEM applications. The board is a complete computer system containing the CPU, system clock, dual port read/write memory, nonvolatile read only memory, I/O ports and drivers, serial communications interface, priority interrupt logic and programmable timers, all on a single 6.75 x 12.00 in. printed circuit card. The iSBC 86/35 board is distinguished by its large RAM content of 512K bytes which is expandable on-board to 1 megabyte; the direct addressing capability of the 8086-2 CPU. The large, on-board memory resource combined with the iAPX 86/10 microprocessor provides high-level system performance ideal for applications, such as robotics, process control, medical instrumentation, office systems, and business data processing.



FUNCTIONAL DESCRIPTION

Overview

The iSBC 86/35 board combines the power of the industry standard 8086 CPU with up to a megabyte page of board resident, dual ported system memory to improve the systems overall performance. By placing the direct memory addressing capability of the iAPX 86/10 CPU on board, MULTIBUS® access to system memory can be eliminated, significantly improving system throughput. Intel's incorporation of 256K bit DRAM technology, parallel and serial I/O, iSBX™ connectors, and interrupt control capabilities make this high performance single board computer system a reality.

Central Processing Unit

The central processor for the iSBC 86/35 board is Intel's iAPX 86/10 (8086-2) CPU. A clock rate of 8 MHz is supported with a jumper selectable option for 5 MHz. The CPU architecture includes four 16-bit byte addressable data registers, two 16-bit index registers, all accessed by a total of 24 operand addressing modes for comprehensive memory addressing and for support of the data structures required for today's structured, high level languages as well as assembly language.

Instruction Set

The 8086 instruction repertoire includes variable length instruction format (including double operand instructions), 8-bit and 16-bit signed and unsigned arithmetic operators for binary, BCD and unpacked ASCII data, and iterative word and byte string manipulation functions.

For enhanced 5 or 8 MHz numerics processing capability, the iSBC 337 MULTIMODULE Numeric Data Processor extends the iAPX 86/10 architecture and data set. Over 60 numeric instructions offer arithmetic, trigonometric, transcendental, logarithmic and exponential instructions. Supported data types include 16-, 32- and 64-bit integer, and 32- and 64-bit floating point, 18-digit packed BCD and 80-bit temporary.

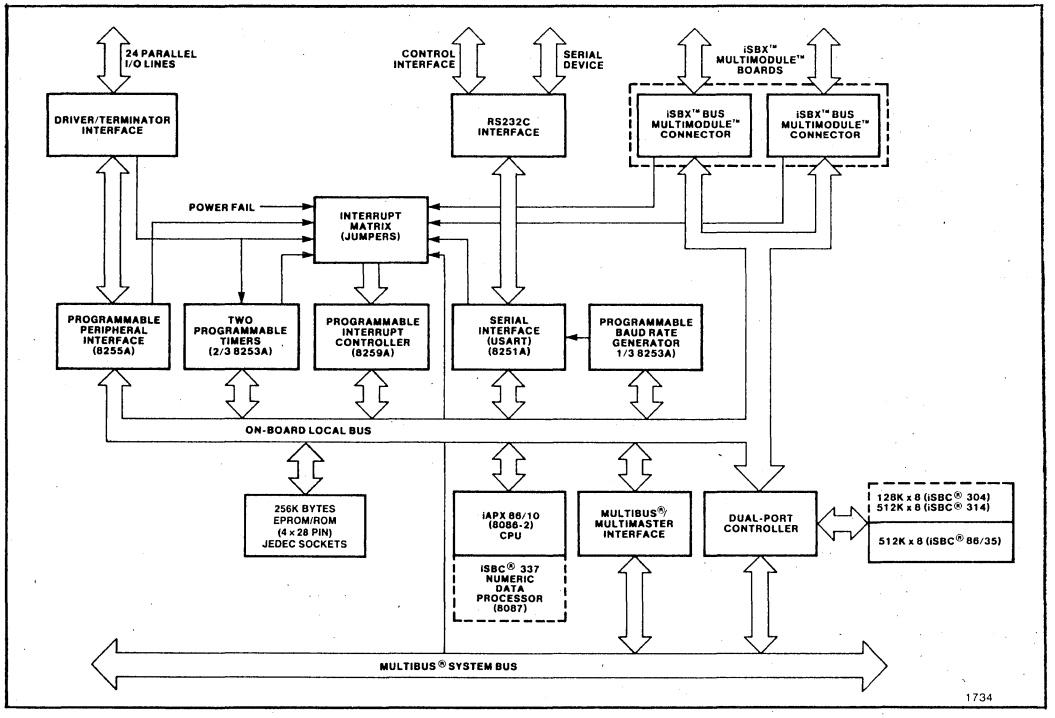


Figure 1. iSBC® 86/35 Block Diagram

Architectural Features

A 6-byte instruction queue provides pre-fetching of sequential instructions and can reduce the 750 nsec minimum instruction cycle to 250 nsec for queued instructions. The stack-oriented architecture readily supports modular programming by facilitating fast, simple, inter-modular communication, and other programming constructs needed for asynchronous real-time systems. The memory expansion capabilities offer a 1 megabyte addressing range. The dynamic relocation scheme allows ease in segmentation of pure procedure and data for efficient memory utilization. Four segment registers (code, stack, data, extra) contain program loaded offset values which are used to map 16-bit addresses to 20-bit addresses. Each register maps 64K bytes at a time and activation of a specific register is controlled explicitly by program control and is also selected implicitly by specific functions and instructions.

RAM Capabilities

The iSBC 86/35 microcomputer contains 512K bytes of dual-port dynamic RAM which may be expanded on-board by adding a RAM Multimodule board as an option. The on-board RAM may be expanded to 640K bytes with the iSBC 304 MULTIMODULE board mounted onto the iSBC 86/35 board. Likewise, the iSBC 86/35 microcomputer may be expanded to 1 Megabyte with the iSBC 314 MULTIMODULE board option.

The dual-port controller allows access to the on-board RAM (including RAM MULTIMODULE board options) from the iSBC 86/35 board and from any other MULTIBUS master via the system bus. Segments of on-board RAM may be configured as a private resource, protected from MULTIBUS system access.

EPROM Capabilities

Four 28-pin JEDEC sockets are provided for the use of Intel 2764, 27128, 27256, 27512, EPROMS and their respective ROMs. When using 27512, the on-board EPROM capacity is 256K bytes. Other JEDEC standard pinout devices are also supported, including byte-wide static RAMs.

Parallel I/O Interface

The iSBC 86/35 Single Board Computer contains 24 programmable parallel I/O lines implemented using the Intel 8255A Programmable Peripheral Interface. The system software is used to configure the I/O lines in any combination of unidirectional input/output and bidirectional ports indicated in Table 1. In order to take advantage of the large number of possible I/O configurations, sockets are provided for interchangeable I/O line drivers and terminators, allowing the selection of the appropriate combination of optional line drivers and terminators with the required drive/termination characteristics. The 24 programmable I/O lines and signal ground lines are brought out to a 50-pin edge connector.

Table 1. Input/Output Port Modes of Operation

Port	Lines (qty)	Mode of Operation					
		Unidirectional				Bidirectional	Control
		Input		Output			
		Latched	Latched & Strobed	Latched	Latched & Strobed		
1	8	X	X	X	X	X	
2	8	X	X	X	X		
3	4	X		X			X[1]
	4	X		X			X[1]

NOTE: 1. Part of port 3 must be used as a control port when either port 1 or port 2 are used as a latched and strobed input or a latched and strobed output port or port 1 is used as a bidirectional port.

Serial I/O

A programmable communications interface using the Intel 8251A Universal Synchronous/Asynchronous Receiver/Transmitter (USART) is contained on the iSBC 86/35 board. A software selectable baud rate generator provides the USART with all common communication frequencies. The mode of operation (i.e., synchronous or asynchronous), data format, control character format, parity, and baud rate are all under program control. The 8251A provides full duplex, double buffered transmit and receive capability. Parity, overrun, and framing error detection are all incorporated in the USART. The RS232C command lines, serial data lines and signal ground line are brought out to a 26-pin edge connector.

Programmable Timers

The iSBC 86/35 board provides three independent, fully programmable 16-bit interval timers/event counters utilizing the Intel 8253 Programmable Interval Timer. Each counter is capable of operating in either BCD or binary modes. Two of these timers/counters are available to the systems designer to generate accurate time intervals under software control. Routing for the outputs and gate/trigger inputs of two of these counters is jumper selectable. The outputs may be independently routed to the 8259A Programmable Interrupt Controller and to the I/O terminators associated with the 8255A to allow external devices or an 8255A port to gate the timer or to count external events. The third interval timer in the 8253 provides the programmable baud rate generator for the iSBC 86/35 board's RS232C USART serial port. The system software configures each timer independently to select the desired function. Seven functions are available as shown in Table 2. The contents of each counter may be read at any time during system operation.

iSBX™ MULTIMODULE™ On-Board Expansion

Two 8/16-bit iSBX MULTIMODULE connectors are provided on the iSBC 86/35 microcomputer. Through these connectors, additional on-board I/O functions may be added. iSBX MULTIMODULE boards optimally support functions provided by VLSI peripheral components such as additional parallel and serial I/O, analog I/O, small mass storage device controllers (e.g., cassettes and floppy disks), and other custom

Table 2. Programmable Timer Functions

Function	Operation
Interrupt on terminal count	When terminal count is reached, an interrupt request is generated. This function is extremely useful for generation of real-time clocks.
Programmable one-shot	Output goes low upon receipt of an external trigger edge or software command and returns high when terminal count is reached. This function is retriggerable.
Rate generator	Divide by N counter. The output will go low for one input clock cycle, and the period from one low going pulse to the next is N times the input clock period.
Square-wave rate generator	Output will remain high until one-half the count has been completed, and go low for the other half of the count.
Software triggered strobe	Output remains high until software loads count (N). N counts after count is loaded, output goes low for one input clock period.
Hardware triggered strobe	Output goes low for one clock period N counts after rising edge counter trigger input. The counter is retriggerable.
Event counter	On a jumper selectable basis, the clock input becomes an input from the external system. CPU may read the number of events occurring after the conter "window" has been enabled or an interrupt may be generated after N events occur in the system.

interfaces to meet specific needs. By mounting directly on the single board computer, less interface logic, less power, simpler packaging, higher performance, and lower cost result when compared to other alternatives such as MULTIBUS form factor compatible boards. The iSBX connectors on the iSBC 86/35 board provides all signals necessary to interface to the local on-board bus, including 16 data lines for maximum data transfer rates. iSBX MULTIMODULE boards designed with 8-bit data paths and using the 8-bit iSBX connector are also supported on the iSBC 86/35 microcomputer. A broad range of iSBX MULTIMODULE options are available in this family from Intel. Custom iSBX modules may also be designed for use on the iSBC 86/35 board. An iSBX bus interface specification and iSBX connectors are available from Intel.

MULTIBUS® System Bus Capabilities

Overview

The MULTIBUS system bus is Intel's industry standard (IEEE 796) microcomputer bus structure. Both 8- and 16-bit single board computers are supported on the MULTIBUS structure with 24 address and 16 data lines. In its simplest application, the MULTIBUS system bus allows expansion of functions already contained on a single board computer (e.g., memory and digital I/O). However, the MULTIBUS structure also allows very powerful distributed processing configurations with multiple processors and intelligent slave I/O, and peripheral boards capable of solving the most demanding microcomputer applications. The MULTIBUS system bus is supported with a broad array of board level products, LSI interface components, detailed published specifications and application notes. Please refer to the MULTIBUS Handbook (order number 210883) for more detailed information.

Multimaster Capabilities

For those applications requiring additional processing capacity and the benefits of multiprocessing (i.e. several CPUs and/or controllers logically sharing system tasks through communication on the system bus), the iSBC 86/35 board provides full MULTIBUS arbitration control logic. This control logic allows both serial (daisy chain) and parallel priority schemes. The serial scheme allows up to three iSBC 86/35 boards/bus masters to share the MULTIBUS system bus; while up to 16 masters may be connected using the parallel scheme and external decode logic.

Interrupt Capability

The iSBC 86/35 board provides 9 vectored interrupt levels. The highest level is the NMI (Non-Maskable Interrupt) line which is directly tied to the 8086-2 CPU. This interrupt is typically used for signaling catastrophic events (e.g., power failure). The Intel 8259A Programmable Interrupt Controller (PIC) provides control and vectoring for the next eight interrupt levels. As shown in Table 3, a selection of four priority processing modes is available for use in designing request processing configurations to match system requirements for efficient interrupt servicing with minimal latencies. Operating mode and priority assignments may be reconfigured dynamically via software at any time during system operation. The PIC accepts interrupt requests from all on-board I/O resources and from the MULTIBUS system bus. The PIC then resolves requests according to the selected mode and, if appropriate, issues an interrupt to the CPU. Any combination of interrupt levels may be masked via software, by storing a single byte in the interrupt mask register of the PIC. In systems requiring additional interrupt levels, slave 8259A PICs may be interfaced via the MULTIBUS system bus, to generate additional vector addresses, yielding a total of 65 unique interrupt levels.

Table 3. Programmable Interrupt Modes

Mode	Operation
Fully nested	Interrupt request line priorities fixed at 0 as highest, 7 as lowest.
Auto-rotating	Equal priority. Each level, after receiving service, becomes the lowest priority level until next interrupt occurs.
Specific priority	System software assigns lowest priority level. Priority of all other levels based in sequence numerically on this assignment.
Polled	System software examines priority-encoded system interrupt status via interrupt status register.

Interrupt Request Generation

Interrupt requests to be serviced by the iSBC 86/35 board may originate from 28 sources. Table 4 includes a list of devices and functions supported by interrupts. All interrupt signals are brought to the interrupt jumper matrix where any combination of interrupt sources may be strapped to the desired interrupt request level on the 8259A PIC or the NMI input to the CPU directly.

Power-Fail Control and Auxiliary Power

Control logic is included to accept a power-fail interrupt in conjunction with the AC-low signal from the Power Supply to initiate an orderly shut down of the system in the event of a power failure. Additionally, an active-low TTL compatible memory protect signal is brought out on the auxiliary connector which, when asserted, disables read/write access to RAM memory on the board. This input is provided for the protection of RAM contents during system power-down

Table 4. Interrupt Request Sources

Device	Function	Number of Interrupts
MULTIBUS® interface	Requests from MULTIBUS® resident peripherals or other CPU boards	8; may be expanded to 64 with slave 8259A PICs on MULTIBUS® boards
8255A Programmable Peripheral Interface	Signals input buffer full or output buffer empty; also BUS INTR OUT general purpose interrupt from driver/terminator sockets	3
8251A USART	Transmit buffer empty and receive buffer full	2
8253 Timers	Timer 0, 1 outputs; function determined by timer mode	2
iSBX™ connectors	Function determined by iSBX™ MULTIMODULE™ board	4 (2 per iSBX™ connector)
Bus fail safe timer	Indicates addressed MULTIBUS® resident device has not responded to command within 6 msec	1
Power fail interrupt	Indicates AC power is not within tolerance	1
Power line clock	Source of 120 Hz signal from power supply	1
External interrupt	General purpose interrupt from auxiliary (P2) connector on backplane	1
iSBC® 337 MULTIMODULE™ Numeric Data Processor	Indicates error or exception condition	1
Edge-level conversion	Converts edge triggered interrupt request to level interrupt	1
OR-gate matrix	Outputs of OR-gates on-board for multiple interrupts	2

sequences. An auxiliary power bus is also provided to allow separate power to RAM for systems requiring battery backup of read/write memory. Selection of this auxiliary RAM power bus is made via jumpers on the board.

System Development Capabilities

The development cycle of iSBC 86/35 products can be significantly reduced and simplified by using either the System 86/330 or the Intellec® Series Microcomputer Development Systems. The Assembler, Locating Linker, Library Manager, Text Editor and System Monitor are all supported by the ISIS-II disk-based operating system.

IN-CIRCUIT EMULATOR

The Intellec ICE™-86A In-Circuit Emulator provides the necessary link between the software development environment provided by the Intellec system and the "target" iSBC 86/35 execution system. In addition to providing the mechanism for loading executable code and data into the iSBC 86/35 board, the ICE-86A In-Circuit Emulator provides a sophisticated command set to assist in debugging software and final integration of the user hardware and software.

Software Support

Real time support for the iSBC 86/35 board is provided by the iRMX 86 operating system. The iRMX 86 Operating System is a highly functional operating system with a rich set of features and options based on an object-oriented architecture. In addition to being modular and configurable, functions beyond the nucleus include a sophisticated file management and I/O system, and powerful human interface. Both packages are easily customized and extended by the user to match unique requirements.

Interactive multi-user support will be provided by the Xenix* operating system. Xenix is a compatible derivative of Unix**, System III.

Language support for the iSBC 86/35 board includes Intels ASM 86, PL/M 86, and PASCAL, and FORTRAN, as well as many third party 8086 languages. Programs developed in these languages can be downloaded from an Intel Series II, or Series III development system to the iSBC 86/35 board via the iSDM™ 86 system debug monitor. The iSDM 86 monitor also provides on-target, interactive system debug capability including breakpoint and memory examination features. The monitor supports iSBC/iAPX 86, 88, 186, and 188 based applications.

* is a trademark of Microsoft Corp.
** is a trademark of Bell Labs.

SPECIFICATIONS

Word Size

INSTRUCTION — 8, 16, 24, or 32 bits

DATA — 8, 16 bits

System Clock

5 MHz or 8 MHz ± 0.1% (jumper selectable)

Cycle Time

BASIC INSTRUCTION CYCLE

8 MHz — 250 ns (assumes instruction in the queue)
5 MHz — 400 ns (assumes instruction in the queue)

NOTE: Basic instruction cycle is defined as the fastest nstruction time (i.e., two clock cycles).

Memory Capacity/Addressing

ON-BOARD EPROM

Device	Total Capacity	Address Range
2764	32K bytes	$F8000\text{-}FFFFF_H$
27128	64K bytes	$F0000\text{-}FFFFF_H$
27256	128K bytes	$E0000\text{-}FFFFF_H$
27512	256K bytes	$D0000\text{-}FFFFF_H$

ON-BOARD RAM

Board	Total Capacity	Address Range
iSBC 86/35	512K bytes	$0\text{-}7FFFF_H$

WITH MULTIMODULE™ RAM

Board	Total Capacity	Address Range
iSBC 304	640K bytes	$8\text{-}9\ FFFF_H$
iSBC 314	1M bytes	$8\text{-}FFFFF_H$

I/O Capacity

PARALLEL — 24 programmable lines using one 8255A.

SERIAL — 1 programmable line using one 8251A

iSBX™ MULTIMODULE™ — 2 iSBX boards

Serial Communications Characteristics

SYNCHRONOUS — 5-8 bit characters; internal or external character synchronization; automatic sync insertion

ASYNCHRONOUS — 5-8 bit characters; break character generation; 1, 1½, or 2 stop bits; false start bit detection

BAUD RATES

Frequency (kHz) (Software Selectable)	Baud Rate (Hz)		
	Synchronous	Asynchronous	
		÷16	÷64
153.6	—	9600	2400
76.8	—	4800	1200
38.4	38400	2400	600
19.2	19200	1200	300
9.6	9600	600	150
4.8	4800	300	75
2.4	2400	150	—
1.76	1760	110	—

NOTE: Frequency selected by I/O write of appropriate 16-bit frequency factor to baud rate register (8253 Timer 2).

Timers

INPUT FREQUENCIES

Reference: 2.46 MHz ±0.1% (0.041 μsec period, nominal); or 153.60 kHz ±0.1% (6.51 μsec period, nominal)

NOTE: Above frequencies are user selectable.

Event Rate: 2.46 MHz max

OUTPUT FREQUENCIES/TIMING INTERVALS

Function	Single Timer/Counter		Dual Timer/Counter (Cascaded)	
	Min	Max	Min	Max
Real-time Interrupt	1.63μs	427.1 ms	3.26s	466.50 min
Programmable one-shot	1.63μs	427.1 ms	3.26s	466.50 min
Rate generator	2.342 Hz	613.5 kHz	0.000036 Hz	306.8 kHz
Square-wave rate generator	2.342 Hz	613.5 kHz	0.oooo36 Hz	306.8 kHz
Software triggered strobe	1.63 μs	427.1 ms	3.26s	466.50 min
Hardware triggered strobe	1.63 μs	427.1 ms	3.26s	466.50 min
Event counter	—	2.46 MHz	—	—

Interfaces

MULTIBUS® — All signals TTL compatible

iSBX™ BUS — All signals TTL compatible

PARALLEL I/O — All signals TTL compatible

SERIAL I/O — RS232C compatible, configurable as a data set or data terminal

TIMER — All signals TTL compatible

INTERRUPT REQUESTS — All TTL compatible

Connectors

Interface	Double-Sided Pins	(in.)	Connectors
MULTIBUS® System	86	0.156	Viking 3KH43/9AMK12 Wire Wrap
iSBX™ Bus			Viking
8-Bit Data	36	0.1	000292-0001
16-Bit Data	44	0.1	000293-0001
Parallel I/O (2)	50	0.1	3M3415-000 Flat or TI H312125 Pins
Serial I/O	26	0.1	3M 3462-0001 Flat or AMP 88106-1 Flat

Line Drivers and Terminators

I/O DRIVERS — The following line drivers are all compatible with the I/O driver sockets on the iSBC 86/05 board

Driver	Characteristic	Sink Current (mA)
7438	I,OC	48
7437	I	48
7432	NI	16
7426	I,OC	16
7409	NI,OC	16
7408	NI	16
7403	I,OC	16
7400	I	16

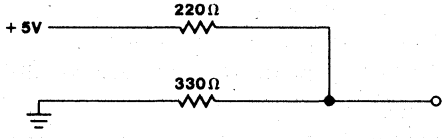
NOTE: I= inverting; NI= non-inverting; OC= open collector.

Port 1 of the 8255A has 20 mA totem-pole bi-directional drives and 1 kΩ terminators

I/O TERMINATORS — 220Ω/330Ω divider or 1 kΩ pullup

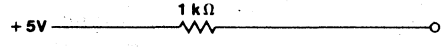
(OPTION 1)

220Ω/330Ω



(OPTION 2)

1 kΩ



MULTIBUS® Drivers

Function	Characteristic	Sink Current (mA)
Data	Tri-State	32
Address	Tri-State	32
Commands	Tri-State	32
Bus Control	Open Collector	20

Physical Characteristics

WIDTH — 12.00 in. (30.48 cm)
HEIGHT — 6.75 in. (17.15 cm)
DEPTH — 0.70 in. (1.78 cm)
WEIGHT — 14 oz. (388 gm)

Electrical Characteristics

DC POWER REQUIREMENTS

Configuration	Current Requirements (All Voltages ±5%)		
	+5V	+12V	−12V
Without EPROM[1]	5.1A	25 mA	23 mA
RAM only[2]	600 mA	—	—
With 32K EPROM[3] (using 2764)	5.6A	25 mA	23 mA
With 64K EPROM (using 27128)	5.7A	25 mA	23 mA
With 128K EPROM (using 27256)	5.8A	25 mA	23 mA

NOTES:
1. Does not include power for optional ROM/EPROM, I/O drivers, and I/O terminators.
2. RAM chips powered via auxiliary power bus in power-down mode.
3. Includes power required for 4 ROM/EPROM chips, and I/O terminators installed for 16 I/O lines; all terminator inputs low.

Environmental Characteristics

OPERATING TEMPERATURE — 0°C to 55°C @ 200 linear feet per minute (LFM) air velocity

RELATIVE HUMIDITY — to 90% (without condensation)

Reference Manual

146245-001 — iSBC 86/35 Hardware Reference Manual (NOT SUPPLIED)

Manuals may be ordered from any Intel sales representative, distributor office of from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

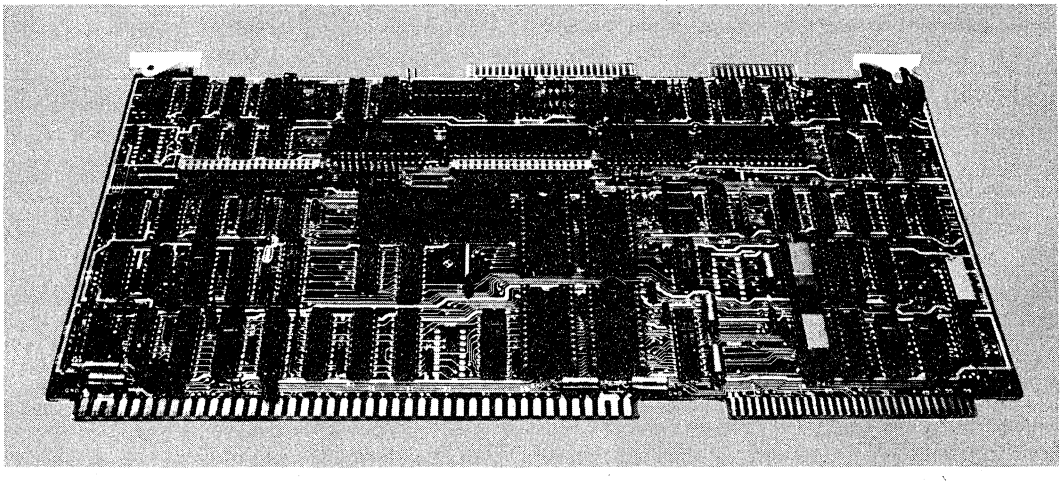
ORDERING INFORMATION

Part Number	Description
SBC 86/35	Single Board Computer

iSBC® 88/25
SINGLE BOARD COMPUTER

- 8-bit 8088 Microprocessor operating at 5 MHz
- One megabyte addressing range
- Two iSBX™ bus connectors
- Optional Numeric Data Processor with iSBC® 337 MULTIMODULE™ Processor
- 4K bytes of static RAM; expandable on-board to 16K bytes
- Sockets for up to 64K bytes of JEDEC 24/28-pin standard memory devices; expandable on-board to 128K bytes
- Programmable synchronous/asynchronous RS232C compatible serial interface with software selectable baud rates
- 24 programmable parallel I/O lines
- Two programmable 16-bit BCD or binary timers/event counters
- 9 Levels of vectored interrupt control, expandable to 65 levels
- MULTIBUS® interface for multimaster configurations and system expansion
- Development support with Intel's iPDS, low cost Personal Development System, and EMV-88 Emulator

The iSBC 88/25 Single Board Computer is a member of Intel's complete line of OEM microcomputer systems which take full advantage of Intel's technology to provide economical, self-contained, computer-based solutions for OEM applications. The iSBC 88/25 board is a complete computer system on a single 6.75 × 12.00-in. printed circuit card. The CPU, system clock, read/write memory, nonvolatile read only memory, I/O ports and drivers, serial communications interface, priority interrupt logic and programmable timers, all reside on the board. The large control storage capacity makes the iSBC 88/25 board ideally suited for control-oriented applications such as process control, instrumentation, industrial automation, and many others.



October, 1984
Order Number: 143847-002

FUNCTIONAL DESCRIPTION

Central Processing Unit

The central processor for the iSBC 88/25 board is Intel's 8088 CPU operating at 5 MHz. The CPU architecture includes four 16-bit byte addressable data registers, two 16-bit memory base pointer registers and two 16-bit index registers, all accessed by a total of 24 operand addressing modes for comprehensive memory addressing and for support of the data structures required for today's structured, high level languages, as well as assembly language.

Instruction Set

The 8088 instruction repertoire includes variable length instruction format (including double operand instructions), 8-bit and 16-bit signed and unsigned arithmetic operators for binary, BCD and unpacked ASCII data, and iterative word and byte string manipulation functions.

For enhanced numerics processing capability, the iSBC 337 MULTIMODULE Numeric Data Processor extends the architecture and data set. Over 60 numeric instructions offer arithmetic, trigonometric, transcendental, logarithmic and exponential instructions. Supported data types include 16, 32, and 64-bit integer, and 32 and 64-bit floating point, 18-digit packed BCD and 80-bit temporary.

Architectural Features

A 4-byte instruction queue provides pre-fetching of sequential instructions and can reduce the 750 nsec minimum instruction cycle to 250 nsec for queued instructions. The stack-oriented architecture readily supports modular programming by facilitating fast, simple, inter-module communication, and other programming constructs needed for asynchronous real-time systems. The memory expansion capabilities offer a 1 megabyte addressing range. The dynamic relocation scheme allows ease in segmentation of pure procedure and data for efficient memory utilization. Four segment registers (code, stack, data, extra) contain program loaded offset values which are used to map 16-bit addresses to 20-bit addresses. Each register maps 64K bytes at a time and activation of a specific register is controlled explicitly by program control and is also selected implicitly by specific functions and instructions. All Intel® languages support the extended memory capability, relieving the programmer of managing the megabyte memory space, yet allowing explicit control when necessary.

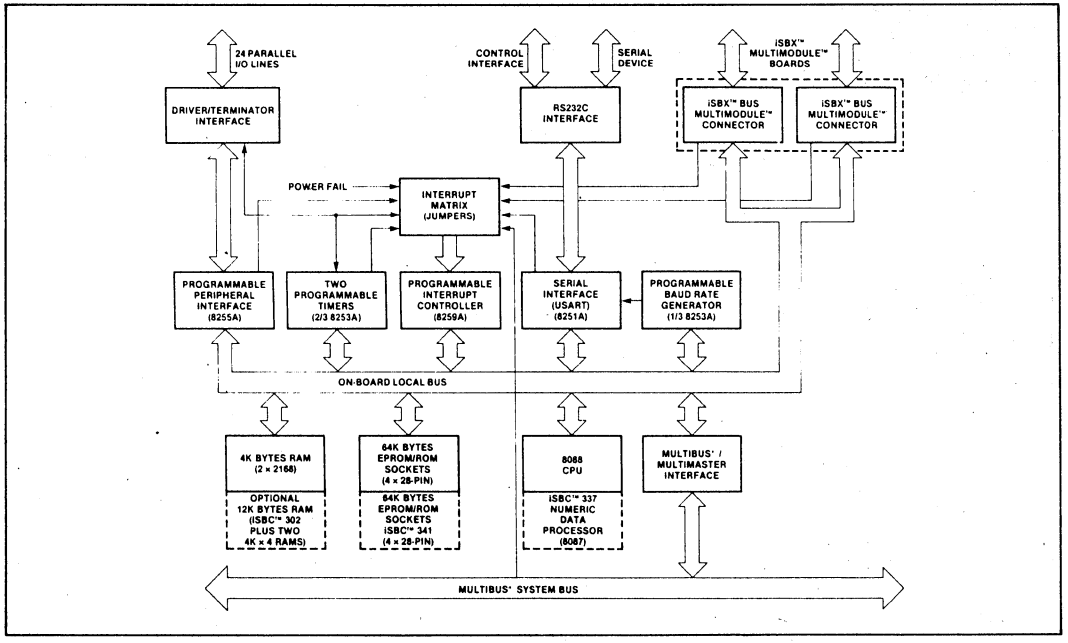


Figure 1. iSBC ® 88/25 Block Diagram

Memory Configuration

The iSBC 88/25 microcomputer contains 4K bytes of high-speed static RAM on-board. In addition, the on-board RAM may be expanded to 12K bytes via the iSBC 302 8K byte RAM module which mounts on the iSBC 88/25 board and then to 16K bytes by adding two 4K × 4 RAM devices in sockets on the iSBC 302 module. All on-board RAM is accessed by the 8088 CPU with no wait states, yielding a memory cycle time of 800 nsec.

In addition to the on-board RAM, the iSBC 88/25 board has four 28-pin sockets, configured to accept JEDEC 24/28-pin standard memory devices. Up to 64K bytes of EPROM are supported in 16K-byte increments with Intel 27128 EPROMs. The iSBC 88/25 board is also compatible with the 2716, 2732, and 2764 EPROMs allowing a capacity of 8K, 16K, and 32K bytes, respectively. Other JEDEC standard pinout devices are also supported, including byte-wide static and integrated RAMs.

With the addition of the iSBC 341 MULTIMODULE EPROM option, the on-board capacity for these devices is doubled, providing up to 128K bytes of EPROM capacity on-board.

Parallel I/O Interface

The iSBC 88/25 Single Board Computer contains 24 programmable parallel I/O lines implemented using the Intel 8255A Programmable Peripheral Interface. The system software is used to configure the I/O lines in any combination of unidirectional input/output and bidirectional ports indicated in Table 1. In order to take advantage of the large number of possible I/O configurations, sockets are provided for interchangeable I/O line drivers and terminators, allowing the selection of the appropriate combination of optional line drivers and terminators with the required drive/termination characteristics. The 24 programmable I/O lines and signal ground lines are brought out to a 50-pin edge connector.

Serial I/O

A programmable communications interface using the Intel 8251A Universal Synchronous/Asynchronous Receiver/Transmitter (USART) is contained on the iSBC 88/25 board. A software selectable baud rate generator provides the USART with all common communication frequencies. The mode of operation (i.e., synchronous or asynchronous), data format, control character format, parity, and baud rate are all under program control. The 8251A provides full duplex, double buffered transmit and receive capability. Parity, overrun, and framing error detection are all incorporated in the USART. The RS232C compatible interface on each board, in conjunction with the USART, provides a direct interface to RS232C compatible terminals, cassettes, and asynchronous and synchronous modems. The RS232C command lines, serial data lines and signal ground line are brought out to a 26-pin edge connector.

Programmable Timers

The iSBC 88/25 board provides three independent, fully programmable 16-bit interval timers/event counters utilizing the Intel 8253 Programmable

Table 1. Input/Output Port Modes of Operation

Port	Lines (qty)	Mode of Operation					Control
		Unidirectional				Bidirectional	
		Input		Output			
		Latched	Latched & Strobed	Latched	Latched & Strobed		
1	8	X	X	X	X	X	
2	8	X	X	X	X		
3	4	X		X			X[1]
	4	X		X			X[1]

NOTE:
1. Part of port 3 must be used as a control port when either port 1 or port 2 are used as a latched and strobed input or a latched and strobed output port or port 1 is used as a bidirectional port.

Interval Timer. Each counter is capable of operating in either BCD or binary modes. Two of these timers/counters are available to the systems designer to generate accurate time intervals under software control. Routing for the outputs and gate/trigger inputs of two of these counters is jumper selectable. The outputs may be independently routed to the 8259A Programmable Interrupt Controller and to the I/O terminators associated with the 8255A to allow external devices or an 8255A port to gate the timer or to count external events. The third interval timer in the 8253 provides the programmable baud rate generator for the iSBC 88/25 board RS232C USART serial port. The system software configures each timer independently to select the desired function. Seven functions are available as shown in Table 2. The contents of each counter may be read at any time during system operation.

Table 2. Programmable Timer Functions

Function	Operation
Interrupt on terminal count	When terminal count is reached, an interrupt request is generated. This function is extremely useful for generation of real-time clocks.
Programmable one-shot	Output goes low upon receipt of an external trigger edge or software command and returns high when terminal count is reached. This function is retriggerable.
Rate generator	Divide by N counter. The output will go low for one input clock cycle, and the period from one low going pulse to the next is N times the input clock period.
Square-wave rate generator	Output will remain high until one-half the count has been completed, and go low for the other half of the count.
Software triggered strobe	Output remains high until software loads count (N). N counts after count is loaded, output goes low for one input clock period.
Hardware triggered strobe	Output goes low for one clock period N counts after rising edge counter trigger input. The counter is retriggerable.
Event counter	On a jumper selectable basis, the clock input becomes an input from the external system. CPU may read the number of events occurring after the counter "window" has been enabled or an interrupt may be generated after N events occur in the system.

iSBX™ MULTIMODULE™ On-Board Expansion

Two 8-bit iSBX MULTIMODULE connectors are provided on the iSBC 88/25 microcomputer. Through these connectors, additional on-board I/O functions may be added. iSBX MULTIMODULES optimally support functions provided by VLSI peripheral components such as additional parallel and serial I/O, analog I/O, small mass storage device controllers (e.g., cassettes and floppy disks), and other custom interfaces to meet specific needs. By mounting directly on the single board computer, less interface logic, less power, simpler packaging, higher performance, and lower cost result when compared to other alternatives such as MULTIBUS form factor compatible boards. The iSBX connectors on the iSBC 88/25 provide all signals necessary to interface to the local on-board bus. A broad range of iSBX MULTIMODULE options are available in this family from Intel. Custom iSBX modules may also be designed for use on the iSBC 88/25 board. An iSBX bus interface specification and iSBX connectors are available from Intel.

MULTIBUS® SYSTEM BUS AND MULTIMASTER CAPABILITIES

Overview

The MULTIBUS system bus is Intel's industry standard microcomputer bus structure. Both 8 and 16-bit single board computers are supported on the MULTIBUS structure with 24 address and 16 data lines. In its simplest application, the MULTIBUS system bus allows expansion of functions already contained on a single board computer (e.g., memory and digital I/O). However, the MULTIBUS structure also allows very powerful distributed processing configurations with multiple processors and intelligent slave I/O, and peripheral boards capable of solving the most demanding microcomputer applications. The MULTIBUS system bus is supported with a broad array of board level products, LSI interface components, detailed published specifications and application notes.

Expansion Capabilities

Memory and I/O capacity may be expanded and additional functions added using Intel MULTIBUS compatible expansion boards. Memory may be expanded by adding user specified combinations

of RAM boards, EPROM boards, or combination boards. Input/output capacity may be added with digital I/O and analog I/O expansion boards. Mass storage capability may be achieved by adding single or double density diskette controllers, or hard disk controllers. Modular expandable backplanes and cardcages are available to support multiboard systems.

Multimaster Capabilities

For those applications requiring additional processing capacity and the benefits of multiprocessing (i.e., several CPUs and/or controllers logically sharing system tasks through communication of the system bus), the iSBC 88/25 board provides full MULTIBUS arbitration control logic. This control logic allows up to three iSBC 88/25 boards or other bus masters, including iSBC 80 and iSBC 86 family MULTIBUS compatible single board computers to share the system bus using a serial (daisy chain) priority scheme and allows up to 16 masters to share the MULTIBUS system bus with an external parallel priority decoder. In addition to the multiprocessing configurations made possible with multimaster capability, it also provides a very efficient mechanism for all forms of DMA (Direct Memory Access) transfers.

Interrupt Capability

The iSBC 88/25 board provides 9 vectored interrupt levels. The highest level is the NMI (Non-Maskable Interrupt) line which is directly tied to the 8088 CPU. This interrupt is typically used for signaling catastrophic events (e.g., power failure). The Intel 8259A Programmable Interrupt Controller (PIC) provides control and vectoring for the next eight interrupt levels. As shown in Table 3, a selection of four priority processing modes is available for use in designing request processing configurations to match system requirements for efficient interrupt servicing with minimal latencies. Operating mode and priority assignments may be reconfigured dynamically via software at any time during system operation. The PIC accepts interrupt requests from all on-board I/O resources and from the MULTIBUS system bus. The PIC then resolves requests according to the selected mode and, if appropriate, issues an interrupt to the CPU. Any combination of interrupt levels may be masked via software, by storing a single byte in the interrupt mask register of the PIC. In systems requiring additional interrupt levels, slave 8259A PICs may be interfaced via the MULTIBUS system bus, to generate additional vector addresses, yielding a total of 65 unique interrupt levels.

Table 3. Programmable Interrupt Modes

Mode	Operation
Fully nested	Interrupt request line priorities fixed at 0 as highest, 7 as lowest.
Auto-rotating	Equal priority. Each level, after receiving service, becomes the lowest priority level until next interrupt occurs.
Specific priority	System software assigns lowest priority level. Priority of all other levels based in sequence numerically on this assignment.
Polled	System software examines priority-encoded system interrupt status via interrupt status register.

Interrupt Request Generation

Interrupt requests to be serviced by the iSBC 88/25 board may originate from 24 sources. Table 4 includes a list of devices and functions supported by interrupts. All interrupt signals are brought to the interrupt jumper matrix where any combination of interrupt sources may be strapped to the desired interrupt request level on the 8259A PIC or the NMI input to the CPU directly.

Power-Fail Control and Auxiliary Power

Control logic is also included to accept a power-fail interrupt in conjunction with the AC-low signal from the iSBC 635 and iSBC 640 Power Supply or equivalent, to initiate an orderly shut down of the system in the event of a power failure. Additionally, an active-low TTL compatible memory protect signal is brought out on the auxiliary connector which, when asserted, disables read/write access to RAM memory on the board. This input is provided for the protection of RAM contents during system power-down sequences. An auxiliary power bus is also provided to allow separate power to RAM for systems requiring battery back-up of read/write memory. Selection of this auxiliary RAM power bus is made via jumpers on the board.

System Development Capabilities

The development cycle of iSBC 88/25 products can be significantly reduced and simplified by using the Intellec Series Microcomputer Development Systems. The Assembler, Locating Linker, Library Manager, Text Editor and System Monitor

are all supported by the ISIS-II disk-based operating system. To facilitate conversion of 8080A/8085A assembly language programs to run on the iSBC 88/25 board, CONV-86 is available under the ISIS-II operating system. The iSBC 88/25 board is also supported by Intel's iPDS, Personal Development System. This system provides low cost development support while at the same time providing personal computer capability for the engineer.

IN-CIRCUIT EMULATOR

The ICE-88 In-Circuit Emulator provides the necessary link between the software development environment provided by the Intellec system and the "target" iSBC 88/25 execution system. In addition to providing the mechanism for loading executable code and data into the iSBC 88/25 board, the ICE-88 In-Circuit Emulator provides a sophisticated command set to assist in debugging software and final integration of the user hardware and software. The EMV-88 Emulator, designed for 8088-based product support on the iPDS, provides for a complete development solution at low cost.

PL/M-86

Intel's system's implementation language, PL/M-86, is also available as an Intellec Microcomputer Development System option. PL/M-86 provides the capability to program in algorithmic language and eliminates the need to manage register usage or allocate memory while still allowing explicit control of the system's resources when needed.

Run-Time Support

Intel also offers two run-time support packages; iRMX 88 Realtime Multitasking Executive and the iRMX 86 Operating System. iRMX 88 is a simple, highly configurable and efficient foundation for small, high performance applications. Its multitasking structure establishes a solid foundation for modular system design and provides task scheduling and management, intertask communication and synchronization, and interrupt servicing for a variety of peripheral devices. Other configurable options include terminal handlers, disk file system, debuggers and other utilities. iRMX 86 is a high functional operating system with a very rich set of features and options based on an object-oriented architecture. In addition to being modular and configurable, functions beyond the nucleus include a sophisticated file management and I/O system, and powerful human interface. Both packages are easily customized and extended by the user to match unique requirements.

Table 4. Interrupt Request Sources

Device	Function	Number of Interrupts
MULTIBUS interface	Requests from MULTIBUS resident peripherals or other CPU boards	8; may be expanded to 64 with slave 8259A PIC's on MULTIBUS boards
8255A Programmable Peripheral Interface	Signals input buffer full or output buffer empty; also BUS INTR OUT general purpose interrupt from driver/terminator sockets	3
8251A USART	Transmit buffer empty and receive buffer full	2
8253 Timers	Timer 0, 1 outputs; function determined by timer mode	2
iSBX connectors	Function determined by iSBX MULTIMODULE board	4 (2 per iSBX connector)
Bus fail safe timer	Indicates addressed MULTIBUS resident device has not responded to command within 6 msec	1
Power fail interrupt	Indicates AC power is not within tolerance	1
Power line clock	Source of 120 Hz signal from power supply	1
External interrupt	General purpose interrupt from parallel port J1 connector	1
iSBC 337 MULTIMODULE Numeric Data Processor	Indicates error or exception condition	1

SPECIFICATIONS

Word Size

INSTRUCTION — 8, 16, 24, or 32 bits
DATA — 8 bits

System Clock

5.00 MHz or 4.17 MHz ± 0.1% (jumper selectable)

NOTE: 4.17 MHz required with the optional iSBC 337 module.

Cycle Time

BASIC INSTRUCTION CYCLE

At 5 MHz — 1.2 μsec
— 400 nsec (assumes instruction in the queue)

NOTES: Basic instruction cycle is defined as the fastest instruction time (i.e., two clock cycles).

Memory Cycle Time

RAM — 800 nsec (no wait states)
EPROM — Jumper selectable from 800 nsec to 1400 nsec

Memory Capacity/Addressing

ON-BOARD EPROM

Device	Total Capacity	Address Range
2716	8K bytes	FE000-FFFFF$_H$
2732	16K bytes	FC000-FFFFF$_H$
2764	32K bytes	F8000-FFFFF$_H$
27128	64K bytes	F0000-FFFFF$_H$

WITH iSBC 341 MULTIMODULE EPROM

Device	Total Capacity	Address Range
2716	16K bytes	FC000-FFFFF$_H$
2732	32K bytes	F8000-FFFFF$_H$
2764	64K bytes	F0000-FFFFF$_H$
27128	128K bytes	E0000-FFFFF$_H$

NOTES: iSBC 88/25 EPROM sockets support JEDEC 24/28-pin standard EPROMs and RAMs (2 sockets); iSBC 341 sockets also support E^2PROMs.

ON-BOARD RAM
4K bytes — 0-0FFF$_H$

WITH iSBC 302 MULTIMODULE RAM
12K bytes — 0-2FFF$_H$

WITH iSBC 302 MULTIMODULE BOARD AND TWO 4K x 4 RAM CHIPS
16K bytes — 0-3FFF$_H$

I/O Capacity

PARALLEL — 24 programmable lines using one 8255A
SERIAL — 1 programmable line using one 8251A
iSBX MULTIMODULE — 2 iSBX MULTIMODULE boards

Serial Communications Characteristics

SYNCHRONOUS — 5-8 bit characters; internal or external character synchronization; automatic sync insertion

ASYNCHRONOUS — 5-8 bit characters; break character generation; 1, 1½, or 2 stop bits; false start bit detection

BAUD RATES

Frequency (kHz) (Software Selectable)	Baud Rate (Hz)		
	Synchronous	Asynchronous	
		÷16	÷64
153.6	—	9600	2400
76.8	—	4800	1200
38.4	38400	2400	600
19.2	19200	1200	300
9.6	9600	600	150
4.8	4800	300	75
2.4	2400	150	—
1.76	1760	110	—

NOTES:
Frequency selected by I/O write of appropriate 16-bit frequency factor to baud rate register (8253 Timer 2).

Timers

INPUT FREQUENCIES

Reference: 2.458 MHz ± 0.1% (406.9 nsec period, nominal); or 1.229 MHz ± 0.1% (813.8 nsec period, nominal); or 153.6 kHz ± 0.1% (6.510 μsec period, nominal)

NOTES:
Above frequencies are user selectable.

Event Rate: 2.46 MHz max

OUTPUT FREQUENCIES/TIMING INTERVALS

Function	Single Timer/Counter		Dual Timer/Counter (Two Timers Cascaded)	
	Min	Max	Min	Max
Real-time Interrupt	1.63 μs	427.1 ms	3.26s	466.50 min
Programmable one-shot	1.63 μs	427.1 ms	3.26s	466.50 min
Rate generator	2.342 Hz	613.5 kHz	0.000036 Hz	306.8 kHz
Square-wave rate generator	2.342 Hz	613.5 kHz	0.000036 Hz	306.8 kHz
Software triggered strobe	1.63 μs	427.1 ms	3.26s	466.50 min
Hardware triggered strobe	1.63 μs	427.1 ms	3.26s	466.50 min
Event counter	—	2.46 MHz	—	—

Interfaces

MULTIBUS — All signals TTL compatible
iSBX BUS — All signals TTL compatible
PARALLEL I/O — All signals TTL compatible
SERIAL I/O — RS232C compatible, configurable as a data set or data terminal
TIMER — All signals TTL compatible
INTERRUPT REQUESTS — All TTL compatible

Connectors

Interface	Double-Sided Pins (qty)	Centers (in.)	Mating Connectors
MULTIBUS System	86	0.156	Viking 3KH43/9AMK12 Wire Wrap
iSBX Bus 8-Bit Data	36	0.1	iSBX 960-5
Parallel I/O (2)	50	0.1	3M 3415-000 Flat or TI H312125 Pins
Serial I/O	26	0.1	3M 3462-0001 Flat or AMP 88106-1 Flat

Line Drivers and Terminators

I/O DRIVERS — The following line drivers are all compatible with the I/O driver sockets on the iSBC 88/25 board

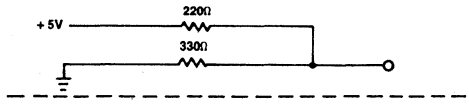
Driver	Characteristic	Sink Current (mA)
7438	I,OC	48
7437	I	48
7432	NI	16
7426	I,OC	16
7409	NI,OC	16
7408	NI	16
7403	I,OC	16
7400	I	16

NOTES:
I = inverting; NI = non-inverting; OC = open collector.

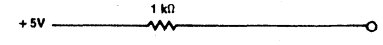
Port 1 of the 8255A has 32 mA totem-pole bidirectional drivers and 10 kΩ terminators

I/O TERMINATORS — 220Ω/330Ω divider or 1 kΩ pullup

220Ω/330Ω Option 1.



1 kΩ Option 2.



MULTIBUS Drivers

Function	Characteristic	Sink Current (mA)
Data	Tri-State	32
Address	Tri-State	24
Commands	Tri-State	32
Bus Control	Open Collector	20

Physical Characteristics

WIDTH — 12.00 in. (30.48 cm)
HEIGHT — 6.75 in. (17.15 cm)
DEPTH — 0.70 in. (1.78 cm)
WEIGHT — 14 oz (388 gm)

Electrical Characteristics

DC POWER REQUIREMENTS

Configuration	Current Requirements (All Voltages ±5%)		
	+5V	+12V	−12V
Without EPROM[1]	3.8A	25 mA	23 mA
RAM only[2]	104 mA		
With 8K EPROM[3] (using 2716)	4.3A	25 mA	23 mA
With 16K EPROM[3] (using 2732)	4.4A	25 mA	23 mA
With 32K EPROM[3] (using 2764)	4.4A	25 mA	23 mA

NOTES:
1. Does not include power for optional ROM/EPROM, I/O drivers, and I/O terminators.
2. RAM chips powered via auxiliary power bus in power-down mode. Does not include power for optional RAM.
3. Includes power required for 4 ROM/EPROM chips, and I/O terminators installed for 16 I/O lines; all terminator inputs low.

Environmental Characteristics

OPERATING TEMPERATURE — 0°C to 55°C
RELATIVE HUMIDITY — to 90% (without condensation)

Reference Manual

143825-001 — iSBC 88/25 Hardware Reference Manual (NOT SUPPLIED)

Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

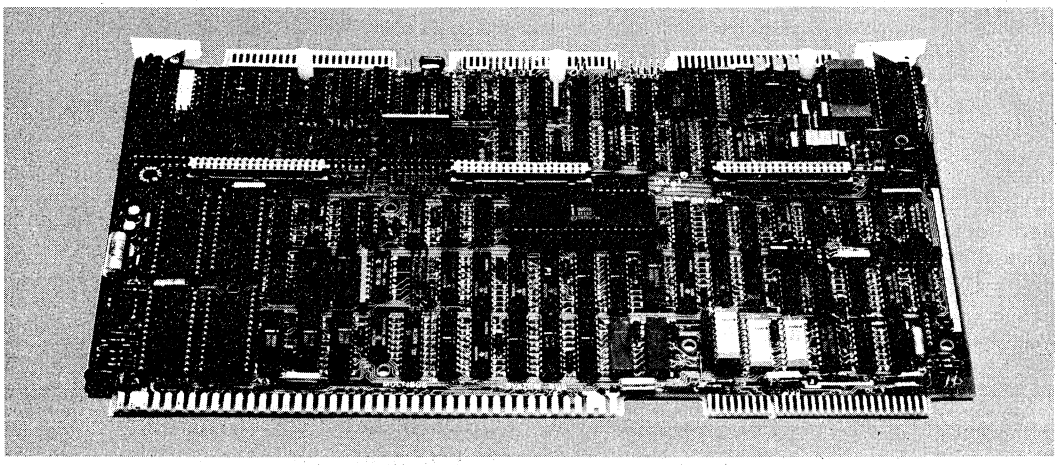
ORDERING INFORMATION

Part Number	Description
SBC 88/25	8-bit Single Board Computer with 4K bytes RAM

iSBC® 88/40
MEASUREMENT AND CONTROL COMPUTER

- **High performance 5 MHz iAPX 88/10 8-bit HMOS processor**
- **12-bit, 20 kHz analog-to-digital converter with programmable gain control**
- **16 differential/32 single-ended analog input channels**
- **Three iSBX™ MULTIMODULE™ connectors for analog, digital, and other I/O expansion**
- **4K bytes static RAM, expandable via iSBC ®301 MULTIMODULE ™RAM to 8K bytes (1K byte dual-ported)**
- **Four EPROM/E^{2}PROM sockets for up to 64K bytes, expandable to 128K bytes with iSBC 341 expansion MULTIMODULE**
- **On-board 21-volt power supply for E^{2}PROM modification under program control**
- **MULTIBUS® Intelligent Slave or Multimaster**

The Intel iSBC 88/40 Measurement and Control Computer is a member of Intel's large family of Single Board Computers that takes full advantage of Intel's VLSI technology to provide an economical self-contained computer based solution for applications in the areas of process control and data acquisition. The on-board iAPX 88/10 processor with its powerful instruction set allows users of the iSBC 88/40 board to update process loops as much as 5-10 times faster than previously possible with other 8-bit microprocessors. For example, the high performance iSBC 88/40 can concurrently process and update 16 control loops in less than 200 milliseconds using a traditional PID (Proportional-Integral-Derivative) control algorithm. The iSBC 88/40 board consists of a 16 differential/32 single ended channel analog multiplexer with input protected circuits, A/D converter, programmable central processing unit, dual port and private RAM, read only memory sockets, interrupt logic, 24 channels of parallel I/O, three programmable timers and MULTIBUS control logic on a single 6.75 by 12.00-inch printed circuit card. The iSBC 88/40 board is capable of functioning by itself in a stand-alone system or as a multimaster or intelligent slave in a large MULTIBUS system.



FUNCTIONAL DESCRIPTION

Three Modes of Operation

The iSBC 88/40 Measurement and Control Computer (MACC) is capable of operating in one of three modes: stand-alone controller, bus multimaster or intelligent slave. A block diagram of the iSBC 88/40 Measurement and Control Computer is shown in Figure 1.

Stand-Alone Controller

The iSBC 88/40 Measurement and Control Computer may function as a stand-alone single board controller with CPU, memory and I/O elements on a single board. The on-board 4K bytes of RAM and up to 64K bytes of read only memory, as well as the analog-to-digital converter and programmable parallel I/O lines allow significant control and monitoring capabilities from a single board.

Bus Multimaster

In this mode of operation the iSBC 88/40 board may interface and control a wide variety of iSBC memory and I/O boards or even with additional iSBC 88/40 boards or other single board computer masters or intelligent slaves.

Intelligent Slave

The iSBC 88/40 board can perform as an intelligent slave to any Intel 8 or 16-bit MULTIBUS master CPU by not only offloading the master of the analog data collection, but it can also do a significant amount of pre-processing and decision making on its own. The distribution of processing tasks to intelligent slaves frees the system master to do other system functions. The dual port RAM with flag bytes for signalling allows the iSBC 88/40 board to process and store data without MULTIBUS memory or bus contention.

Central Processing Unit

The central processor unit for the iSBC 88/40 board is a powerful 8-bit HMOS iAPX 88/10 microprocessor. The 22.5 sq. mil. chip contains approximately 29,000 transistors and has a clock rate of 5 MHz. The architecture includes four (4) addressable data registers and two (2) 16-bit memory base pointer registers and two (2) 16-bit index registers, all accessed by a total of 24 operand addressing modes for complex data handling and very flexible memory addressing.

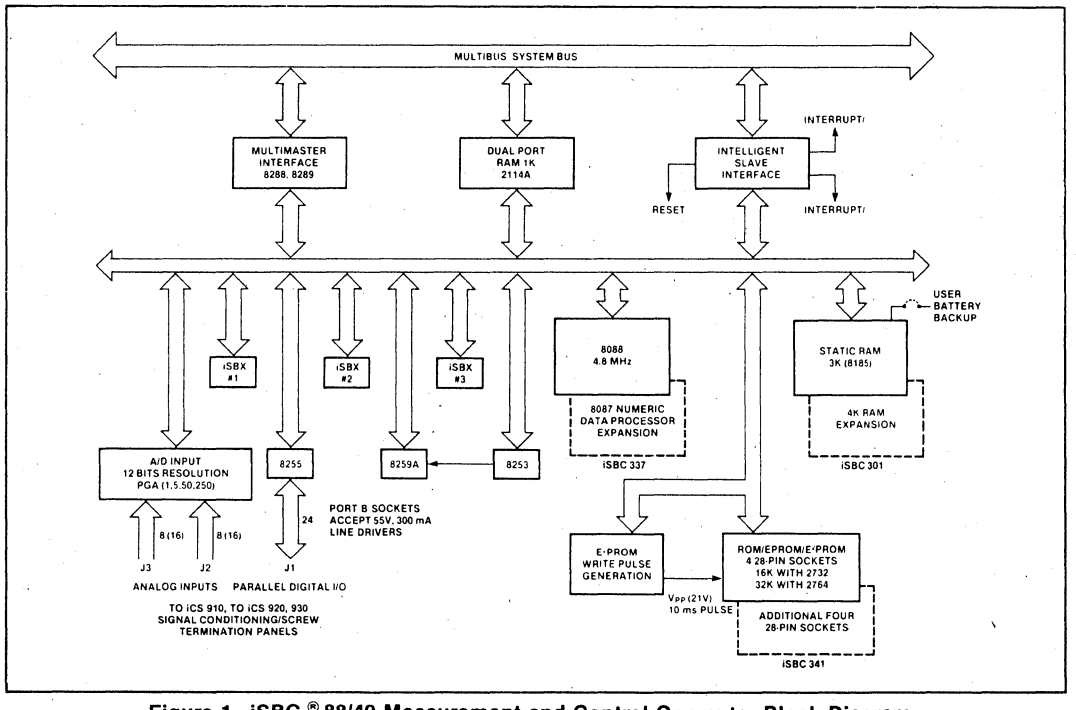


Figure 1. iSBC ® 88/40 Measurement and Control Computer Block Diagram

INSTRUCTION SET — The iAPX 88/10 instruction repertoire includes variable length instruction format (including double operand instructions), 8-bit and 16-bit signed and unsigned arithmetic operators for binary, BCD and unpacked ASCII data, and iterative word and byte string manipulation functions. The instruction set of the iAPX 88/10 is a superset of the 8080A/8085A family and with available software tools, programs written for the 8080A/8085A can be easily converted and run on the iAPX 88/10 processor. Programs can also be run that are implemented on the iAPX 86/10 with little or no modification.

ARCHITECTURAL FEATURES — A 4-byte instruction queue provides pre-fetching of sequential instructions and can reduce the 1.04 msec minimum instruction cycle to 417 nsec for queued instructions. The stack oriented architecture facilitates nested subroutines and co-routines, reentrant code and powerful interrupt handling. The memory expansion capabilities offer a 1 megabyte addressing range. The dynamic relocation scheme allows ease in segmentation of pure procedure and data for efficient memory utilization. Four segment registers (code, stack, data, extra) contain program loaded offset values which are used to map 16-bit addresses to 20-bit addresses. Each register maps 64K bytes at a time and activation of a specific register is controlled explicitly by program control and is also selected implicitly by specific functions and instructions.

Bus Structure

The iSBC 88/40 single board computer has three buses: 1) an internal bus for communicating with on-board memory, analog-to-digital converter, iSBX MULTIMODULES and I/O options; 2) the MULTIBUS system bus for referencing additional memory and I/O options, and 3) the dual-port bus which allows access to RAM from the on-board CPU and the MULTIBUS system bus. Local (on-board) accesses do not require MULTIBUS communication, making the system bus available for use by other MULTIBUS masters (i.e. DMA devices and other single board computers transferring to additional system memory). This feature allows true parallel processing in a multiprocessor environment. In addition, the MULTIBUS interface can be used for system expansion through the use of other 8- and 16-bit iSBC computers, memory and I/O expansion boards.

RAM Capabilities

DUAL-PORT RAM — The dual-port RAM of the iSBC 88/40 board consists of 1K bytes of static RAM, implemented with Intel 2114A chips. The on-board base addess of this RAM is 00C00 (3K) normally; it is relocated to 01C00 (7K) when the iSBC 301 MULTIMODULE RAM is added to the protected RAM. The MULTIBUS port base address of the dual-port RAM can be jumpered to any 1K byte boundary in the 1M byte address space. The dual-port RAM can be accessed in a byte-wide fashion from the MULTIBUS system bus. When accessed from the MULTIBUS system bus, the dual-port RAM decode logic will generate INH1/ (Inhibit RAM) to allow dual-port RAM to overlay other system RAM. The dual-port control logic is designed to favor an on-board RAM access. If the dual-port is not currently performing a memory cycle for the MULTIBUS sytem port, only one wait state will be required. The on-board port may require more than one wait state if the dual-port RAM was busy when the on-board cycle was requested. The LOCK prefix facility of the iAPX 88/10 assembly language will disallow system bus accesses to the dual-port RAM. In addition, the on-board port to the dual-port RAM can be locked by other compatible MULTIBUS masters, which allows true symmetric semaphore operation. When the board is functioning in the master mode, the LOCK prefix will additionally disable other masters from obtaining the system bus.

PRIVATE RAM — In addition to the 1K byte dual-port RAM, there is a 3K byte section of private static RAM not accessible from the system bus. This RAM has a base address of 00000, and consists of three Intel 8185 RAM chips which are interfaced to the multiplexed address/data bus of the iAPX 88/10 microprocessor. Expansion of this private RAM from 3K to 7K bytes can be accomplished by the addition of an iSBC 301 MULTIMODULE RAM (4K bytes). When the 301 is added, protected RAM extends from 0 to 7K, and the base address of the dual-port RAM is relocated from 3K (00C00) to 7K (01C00). All protected RAM accesses require one wait state. The private RAM resides on the local on-board bus, which eliminates contention problems between on-board accesses to private RAM and system bus accesses to dual-port RAM. The private RAM can be battery backed (up to 16K bytes).

Additional RAM can be added by utilizing JEDEC-compatible static RAMs in the available EPROM sockets.

Parallel I/O Interface

The iSBC 88/40 single board computer contains 24 programmable parallel I/O lines implemented using the Intel 8255A Programmable Peripheral In-

terface. The system software is used to configure the I/O lines in any combination of unidirectional input/output and bidirectional ports indicated in Table 1. There the I/O interface may be customized to meet specific peripheral requirements. In order to take full advantage of the large number of possible I/O configurations, sockets are provided for interchangeable I/O line drivers and terminators. Port 2 can also accept TTL compatible peripheral drives, such as 75461/462, 75471/472, etc. These are open collector, high voltage drivers (up to 55 volts) which can sink 300 mA. Hence, the flexibility of the I/O interface is further enhanced by the capability of selecting the appropriate combination of optional line drivers and terminators to provide the required sink current, polarity, and drive/termination characteristics for each application. The 24 programmable I/O lines and signal ground lines are brought out to a 50-pin edge connector that mates with flat, woven, or round cable. This edge connector is also compatible with the Intel iCS 920 Digital I/O and iCS 930 AC Signal Conditioning/Termination Panels, for field wiring, optical isolation and high power (up to 3 amp) power drive.

EPROM Capabilities

Four (4) 28-pin sockets are provided for the use of Intel 2716s, 2732s, 2764s, 27128s, future JEDEC-compatible 128K and 256K bit EPROMs and their respective ROMs. When using 27128s the on-board EPROM capacity is 64K bytes. Read only memory expansion is available through the use of the iSBC 341 EPROM/ROM memory expansion MULTIMODULE. When the iSBC 341 is used an additional four (4) EPROM sockets are made available, for a total iSBC 88/40 board capacity of 128K bytes EPROM with Intel 27128s.

E²PROM Capabilities

The four 28-pin sockets can also accommodate Intel 2816 E²PROMs, for dynamic storage of control loop setpoints, conversion parameters, or other data (or programs) that change periodically but must be kept in nonvolatile storage. To give the user dynamic control of this nonvolatile memory, the iSBC 88/40 board also contains an on-board DC to DC converter which under program control will furnish the voltage necessary for modifying the contents of Intel 2816/2815 E²PROMs.

Timing Logic

The iSBC 88/40 board provides an 8253-5 Programmable Interval Timer, which contains three independent, programmable 16-bit timers/event counters. All three of these counters are available to generate time intervals or event counts under software control. The outputs of the three counters may be independently routed to the interrupt matrix. The inputs and outputs of timers 0 and 1 can be connected to parallel I/O lines on the J1 connector, where they replace 8255A port C lines. The third counter is also used for timing E²PROM write operations.

Interrupt Capability

The iSBC 88/40 board provides 9 vectored interrupt levels. The highest level is the NMI (Non-maskable Interrupt) line which is directly tied to the iAPX 88/10 CPU. This interrupt cannot be inhibited by software and is typically used for signalling catastrophic events (i.e., power failure). On servicing this interrupt, program control will be implicitly transferred through location 00008_H. The Intel 8259A Programmable Interrupt Controller (PIC) provides vectoring for the next eight interrupt levels. As shown in Table 2, a selection

Table 1. Input/Output Port Modes of Operation

Port	Lines (qty)	Mode of Operation					Control
		Unidirectional				Bidirectional	
		Input		Output			
		Latched	Latched & Strobed	Latched	Latched & Strobed		
1	8	X	X	X	X	X	
2	8	X	X	X	X		
3	4	X		X			X[1]
	4	X		X			X[1]

NOTE:

1. Part of port 3 must be used as a control port when either port 1 or port 2 are used as a latched and strobed input or a latched and strobed output port or port 1 is used as a bidirectional port.

of four priority processing modes is available to the designer to match system requirements. Operating mode and priority assignments may be reconfigured dynamically via software at any time during system operation. The PIC accepts interrupt requests from the programmable parallel and/or iSBX interfaces, the programmable timers, the system bus, or directly from peripheral equipment. The PIC then determines which of the incoming requests is of the highest priority, determines whether this request is of higher priority than the level currently being serviced, and, if appropriate, issues an interrupt to the CPU. Any combination of interrupt levels may be masked, via software, by storing a single byte in the interrupt mask register of the PIC. The PIC generates a unique memory address for each interrupt level. These addresses are equally spaced at 4-byte intervals. This 32-byte block may begin at any 32-byte boundary in the lowest 1K bytes of memory*, and contains unique instruction pointers and code segment offset values (for expanded memory operation) for each interrupt level. After acknowledging an interrupt and obtaining a device identifier byte from the 8259A PIC, the CPU will store its status flags on the stack and execute an indirect CALL instruction through the vector location (derived from the device identifier) to the interrupt service routine.

***NOTE:** The first 32 vector locations are reserved by Intel for dedicated vectors. Users who wish to maintain compatibility with present and future Intel products should not use these locations for user-defined vector addresses.

Table 2. Programmable Interrupt Modes

Mode	Operation
Fully nested	Interrupt request line priorities fixed at 0 as highest, 7 as lowest.
Auto-rotating	Equal priority. Each level, after receiving service, becomes the lowest priority level until next interrupt occurs.
Specific priority	System software assigns lowest priority level. Priority of all other levels based in sequence numerically on this assignment.
Polled	System software examines priority-encoded system interrupt status via interrupt status register.

INTERRUPT REQUEST GENERATION — Interrupt requests may originate from 26 sources. Two jumper selectable interrupt requests can be automatically generated by the programmable peripheral interface when a byte of information is ready to be transferred to the CPU (i.e., input buffer is full) or a byte of information has been transferred to a peripheral device (i.e., output buffer is empty). A jumper selectable request can be generated by each of the programmable timers. An additional interrupt request line may be jumpered directly from the parallel I/O driver terminator section. Eight prioritized interrupt request lines allow the iSBC 88/40 board to recognize and service interrupts originating from peripheral boards interfaced via the MULTIBUS system bus. The fail safe timer can be selected as an interrupt source. Also, interrupts are provided from the iSBX connectors (6), end-of-conversion, PFIN and from the power line clock.

Power-Fail Control

Control logic is also included to accept a power-fail interrupt in conjunction with the AC-low signal from the iSBC 635, iSBC 640, and iCS 645 Power Supply or equivalent.

iSBX™ MULTIMODULE™ Expansion Capabilities

Three iSBX MULTIMODULE connectors are provided on the iSBC 88/40 board. Up to three (3) single wide MULTIMODULE or one (1) double wide and one (1) single wide iSBX MULTIMODULE can be added to the iSBC 88/40 board. A wide variety of peripheral controllers, analog and digital expansion options are available. For more information on specific iSBX MULTIMODULES consult the Intel OEM Microcomputer System Configuration Guide.

Processing Expansion Capabilities

The addition of a iSBC 337 Multimodule Numeric Data Processor offers high performance integer and floating point math functions to users of the iSBC 88/40 board. The iSBC 337 incorporates the Intel 8087 and because of the MULTIMODULE implementation, it allows on-board expansion directly on the iSBC 88/40 board, eliminating the need for additional boards for floating point requirements.

MULTIBUS® Expansion

Memory and I/O capacity may be expanded further and additional functions added using Intel MULTIBUS compatible expansion boards. Memory may be expanded by adding user specified combinations of RAM boards, EPROM boards, or memory combination boards. Input/output capacity may be increased by adding digital I/O and analog I/O MULTIBUS expansion boards. Mass storage capability may be acheived by adding single or double density diskette controllers, or hard disk con-

trollers either through the use of expansion boards and iSBX MULTIMODULES. Modular expandable backplanes and cardcages are available to support multiboard systems.

NOTE: Certain system restrictions may be incurred by the inclusion of some of the iSBC 80 family options in an iSBC 88/40 system. Consult the Intel OEM Microcomputer System Configuration Guide for specific data.

Analog Input Section

The analog section of the iSBC 88/40 board receives all control signals through the local bus to initiate channel selection, gain selection, sample and hold operation, and analog-to-digital conversion. See Figure 2.

INPUT CAPACITY — 32 separate analog signals may be randomly or sequentially sampled in single-ended mode with the 32 input multiplexers and a common ground. For noiser environments, differential input mode can be configured to achieve 16 separate differential signal inputs, or 32 pseudo differential inputs.

RESOLUTION — The analog section provides 12-bit resolution with a successive approximation analog-to-digital converter. For bipolar operation (−5 to +5 or −10 to +10 volts) it provides 11 bits plus sign.

SPEED — The A-to-D converter conversion speed is 50 μs (20 kHz samples per second). Combined with the programming interface, maximum throughput via the local bus and into memory will be 55 microseconds per sample, or 18 kHz samples per second, for a single channel, a random channel, or a sequential channel scan at a gain of 1, 5 ms at a gain of 5, 250 ms at a gain of 50, and 20 ms at a gain of 250. A-to-D conversion is initiated via a programmed command from the iAPX 88/10 central processor. Interrupt on end-of-conversion is a standard feature to ease programming and timing constraints.

ACCURACY — High quality components are used to achieve 12 bits resolution and accuracy of 0.035% full scale range ± ½ LSB. Offset is adjustable under program control to obtain a nominal ±0.024% FSR ± ½ LSB accuracy at any fixed temperature between 0°C and 60°C (gain = 1). See specifications for other gain accuracies.

GAIN — To allow sampling of millivolt level signals such as strain gauges and thermocouples, gain is made configurable via user program commands up to 250× (20 millivolts full scale input range). User can select gain ranges of 1 (5V), 5 (1V), 50 (100 mV), 250 (20 mV) to match his application.

OPERATIONAL DESCRIPTION — The iSBC 88/40 single board computer addresses the analog-to-digital converter by executing IN or OUT instructions to the port address. Analog-to-digital conversions can be programmed in either of two modes: 1) start conversion and poll for end-of-conversion (EOC), or 2) start conversion and wait for interrupt at end of conversion. When the conversion is complete as signaled by one of the above techniques, INput instructions read two bytes (low and high bytes) containing the 12-bit data word as shown on the following page.

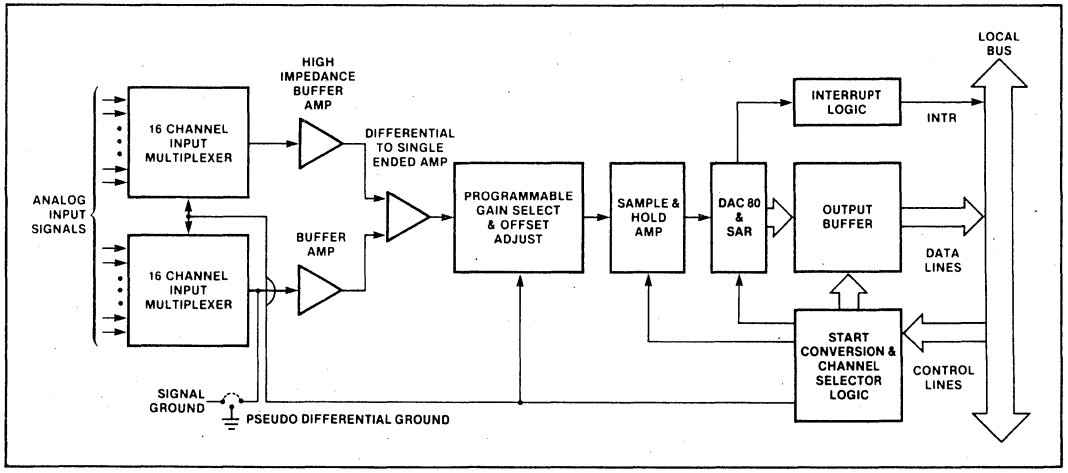


Figure 2. iSBC® 88/40 Analog Input Section

Output Command — Select input channel and start conversion.

	GAIN		CONNECTOR		CHANNEL SELECT			
BIT POSITION	7	6	5	4	3	2	1	0
INPUT CHANNEL	G1	G2		J	C3	C2	C1	C0

Input Data — Read converted data (low byte) or Read converted data (high byte).

	DATA							
BIT POSITION	7	6	5	4	3	2	1	0
LOW/STATUS BYTE	D3	D2	D1	D0				EOC

	DATA							
HIGH BYTE	D11	D10	D9	D8	D7	D6	D5	D4

Offset Correction — At higher gains (×50, ×250) the voltage offset tempco in the A/D circuitry can sometimes cause unacceptable inaccuracies. To correct for this offset, one channel can be dedicated to be used as a reference standard. This channel can be read from the program to determine the amount of offset. The reading from this channel will then be subtracted from all other channel readings, in effect eliminating the offset tempco.

System Software Development

The development cycle of the iSBC 88/40 board may be significantly reduced using an Intel Intellec Microcomputer Development System with the optional iAPX 88/iAPX 86 Software Development package.

The iAPX 88/iAPX 86 Software Development package includes Intel's high-level programming language, PL/M 86. PL/M 86 provides the capability to program in a natural, algorithmic language and eliminates the need to manage register usage or allocate memory. PL/M 86 programs can be written in a much shorter time than assembly language programs for a given application.

SPECIFICATIONS

Word Size

Instruction — 8, 16, or 32 bits

Data — 8 bits

Instruction Cycle Time

417 nanoseconds for fastest executable instruction (assumes instruction is in the queue). 1.04 microseconds for fastest executable instruction (assumes instruction is not in the queue).

Memory Capacity

On-board ROM/EPROM/E²PROM

Up to 64K bytes; user installed in 2K, 4K, 8K or 16K byte increments or up to 128K if iSBC 341 MULTIMODULE EPROM option installed. Up to 8K bytes of E²PROM using Intel 2816s may be user-installed in increments of 2, 4 or 8K bytes.

On-board RAM

4K bytes or 8K bytes if the iSBC 301 MULTIMODULE RAM is installed. Integrity maintained during power failure with user-furnished batteries. 1K bytes are dual-ported.

Off-board Expansion

Up to 1 megabyte of user-specified combination of RAM, ROM, and EPROM.

Memory Addressing

On-board ROM/EPROM

FE000-FFFFF (using 2716 EPROMs)
FC000-FFFFF (using 2732 EPROMs)
F8000-FFFFF (using 2764 EPROMs)
F0000-FFFFF (using 27128 EPROMs)

On-board ROM/EPROM (With iSBC 341 MULTIMODULE EPROM option installed)

FC000-FFFFF (using 2716 EPROMs)
F8000-FFFFF (using 2732 EPROMs)
F0000-FFFFF (using 2764 EPROMs)
E0000-FFFFF (using 27128 EPROMs)

On-board RAM (CPU Access)

00000-00FFF

00000-01FFF (if iSBC 301 MULTIMODULE RAM option installed)

On-board RAM

Jumpers allow 1K bytes of RAM to act as slave RAM for access by another bus master. Addressing may be set within any 1K boundary in the 1-megabyte system address space.

Slave RAM Access

Average; 350 nanoseconds

Interval Timer

Output Frequencies —

Function	Single Timer		Dual Timers (Two Timers Cascaded)
	Min.	Max.	
Real-Time Interrupt Interval	0.977 μs	64 ms	69.9 minutes maximum
Rate Generator (Frequency)	15.625 Hz	1024 kHz	0.00024 Hz minimum

iAPX 88/10 CPU Clock

4.8 MHz ±0.1%

I/O Addressing

All communications to parallel I/O ports, iSBX bus, A/D port, timers, and interrupt controller are via read and write commands from the on-board iAPX 88/10 CPU.

Interface Compatability

Parallel I/O — 24 programmable lines (8 lines per port); one port includes a bidirectional bus driver. IC sockets are included for user installation of line drivers and/or I/O terminators and/or peripheral drivers as required for interface ports.

iSBX Bus Connectors — Three iSBX bus connectors are provided. These connectors accept 8-bit iSBX MULTIMODULE boards. One set of the three iSBX MULTIMODULE connectors will accept a double wide iSBX MULTIMODULE board.

Interrupts

iAPX 88/10 CPU includes a non-maskable interrupt (NMI). NMI interrupt is provided for catastrophic events such as power failure. The on-board 8259A PIC provides 8-bit identifier of interrupting device to CPU. CPU multiplies identifier by four to derive vector address. Jumpers select interrupts from 26 sources without necessity of external hardware. PIC may be programmed to accommodate edge-sensitive or level-sensitive inputs.

Analog Input

16 differential (bipolar operation) or 32 single-ended (unipolar operation).

Full Scale Voltage Range — −5 to +5 volts (bipolar), 0 to +5 volts (unipolar).

NOTE: Ranges of 0 to 10V and ±10V achievable with externally supplied ±15V power.

Gain — Program selectable for gain of 1, 5, 50, or 250.

Resolution — 12 bits (11 bits plus sign for ±5, ±10 volts).

Accuracy — Including noise and dynamic errors

Gain	25°C
1	±0.05% FSR*
5	±0.075% FSR*
50	±0.085% FSR*
250	±0.12% FSR*

* **NOTE:** FSR = Full Scale Range ± ½ LSB. Figures are in percent of full scale reading. At any fixed temperature between 0°C and 60°C, the accuracy is adjustable to ±0.05% of full scale.

Gain TC (at gain = 1) — 30 PPM (typical), 56 PPM (max) per degree centigrade, 40 PPM at other gains.

Offset TC — (in % of FSR/°C)	Gain	Offset TC (typical)
	1	0.0018%
	5	0.0036%
	50	0.024%
	250	0.12%

Sample and Hold-sample Time — 15 μs
Aperature-hold Aperature Time — 120 ns
Input Overvoltage Protection — 30 volts
Input Impedance — 20 megohms (min.)
Conversion Speed — 50 μs (max.) at gain = 1
Common Mode Rejection Ratio — 60 dB (min.)

Physical Characteristics

Width — 30.48 cm (12.00 in.)
Length — 17.15 cm (6.75 in.)
Height — 1.78 cm (0.7 in.)
2.82 cm (1.13 in.) with iSBC Memory Expansion, MULTIMODULES, iSBX Numeric Data Processor or iSBX MULTIMODULES.

Electrical Requirements

Power Requirements (Maximum) —

Configuration	+5V		+5V Aux		+12V		−12V	
	Typ	Max	Typ	Max	Typ	Max	Typ	Max
iSBC 88/40[1,2]	4	5.5	100	150	80	120	30	40

NOTES:
1. The current requirement includes one worst case (active-standby) EPROM current.
2. If +5V Aux is supplied by the iSBC 88/40 board, the total +5V current is the sum of the +5V and the +5V Aux.

Environmental Requirements

Operating Temperature — 0° to 55°C (32°C to 131°F) with 200 lfm air flow
Relative Humidity — to 90% without condensation

Equipment Supplied

The following are supplied with the iSBC 88/40 board:
a. Schematic diagram
b. Assembly drawing

Reference Manuals

124978-001 — iSBC 88/40 Measurement and Control Computer Hardware Reference Manual (NOT SUPPLIED).

Manuals may be ordered from an Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

ORDERING INFORMATION

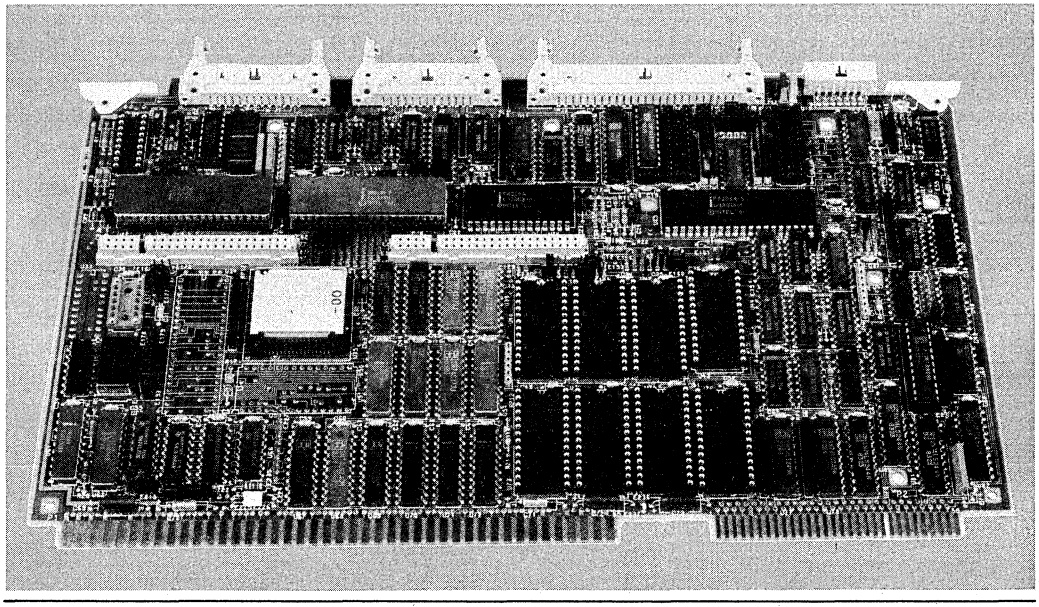
Part Number	Description
SBC 88/40	Measurement and Control Computer

iSBC® 186/03 SINGLE BOARD COMPUTER

- **iSBC® Single Board Computer or mass storage front-end**
- **6.0 MHz iAPX 186/10 (80186) microprocessor**
- **Eight (expandable to 12) JEDEC 28-pin sites**
- **24 programmable I/O lines configurable as a SCSI interface, Centronics interface or general purpose I/O**
- **Two programmable serial interfaces; one RS 232C, the other RS 232C or RS 422 compatible**
- **Six programmable timers**
- **27 levels of vectored interrupt control**
- **80130 real-time Operating System firmware**
- **Two iSBX™ bus interface connectors for low cost I/O expansion**
- **iLBX™ (Local Bus Extension) interface for high-speed memory expansion**
- **MULTIBUS® interface for system expansion and multimaster configuration**

The iSBC® 186/03 Single Board Computer is a member of Intel's complete line of microcomputer modules and systems that take advantage of Intel's VLSI technology to provide economical, off-the-shelf, computer-based solutions for OEM applications. The board is a complete microcomputer system on a 7.05 x 12.0 inch printed circuit card. The CPU, system clock, memory sockets, I/O ports and drivers, serial communications interface, priority interrupt logic and programmable timers, all reside on the board.

The iSBC 186/03 board incorporates the iAPX 186 CPU, 80130 operating system firmware and SCSI interface on one board. The extensive use of high integration VLSI has produced a high-performance single-board system. For large memory applications, the iLBX™ expansion maintains this high performance.



October, 1984
Order Number: 2300988-002

OVERVIEW

Operating Environment

The iSBC 186/03 single board computer features have been designed to meet the needs of numerous microcomputer applications. Typical applications include:

1. Multiprocessing single board computer
2. Mass storage front-end processor
3. Stand-alone Single Board System

MULTIPROCESSING SINGLE BOARD COMPUTER

High-performance systems often need to divide system functions among multiple processors. A multiprocessing Single Board Computer distributes an applications processing load over multiple processors that communicate over a system bus. Since these applications use the system bus for inter-processor communication, it is required that each processor have local execution memory.

The iSBC 186/03 board supports loosely coupled multiprocessing (where each processor performs a specific function) through its MULTIBUS® compatible architecture. The IEEE 796 system bus facilitates processor to processor communication, while the iLBX™ bus makes high-speed data and execution memory available to each CPU as shown in Figure 1. This architecture allows multiple processors to run in parallel enabling very high-performance applications.

MASS STORAGE FRONT-END PROCESSOR

A mass storage front-end processor off-loads a system's central processor of mass storage tasks such as disk control, file buffering and file backup. These applications must support multiple peripherals (tape, floppy, winchester, etc.) and provide versatile CPU pre-processing of disk data.

The iSBC 186/03 board can act as an intelligent disk controller providing both the disk control and the capacity to pre-process disk data. Figure 2 shows the iSBC 186/03 board used as a mass storage front-end processor. The iSBC 186/03 board's SCSI interface and on-board DMA allow high-performance access to SCSI compatible peripherals including Winchester hard disks, floppy disks and cartridge tapes. Data from the SCSI peripherals can be moved directly (using the on-board DMA controller) to on-board memory, iLBX memory or MULTIBUS memory where it can be manipulated by either the 80186 on the iSBC 186/03 board, or by the central processor.

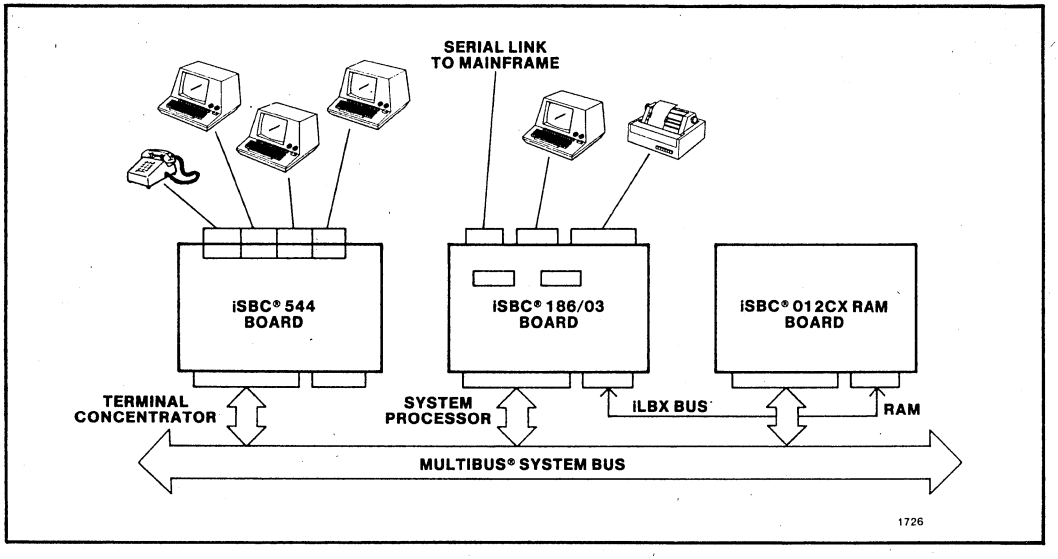


Figure 1. A Multiprocessing Single Board Computer Application

FUNCTIONAL DESCRIPTION

Architecture

The iSBC 186/03 board is functionally partitioned into six major sections: central processor, memory, SCSI compatible parallel interface, serial I/O, interrupt control and MULTIBUS bus expansion. These areas are illustrated in Figure 4.

CENTRAL PROCESSOR

The 80186 central processor along with the 80130 Operating System Firmware component provides high-performance processing power.

The 80186 component is a high-integration 16-bit microprocessor. It combines several of the most common system components onto a single chip (i.e. Direct Memory Access, Interval Timers, Clock Generator and Programmable Interrupt Controller). The 80186 instruction set is a superset of the 8086. It maintains object code compatability while adding ten new instructions. Added instructions include: Block I/O, Enter and Leave subroutines, Push immediate, Multiply Quick, Array Bounds Checking, Shift and Rotate by Immediate, and Pop and Push All.

The 80130 component provides the iRMX™ 86 nucleus primitives for those applications requiring a real-time executive, as well as providing timers and programmable interrupt control. To

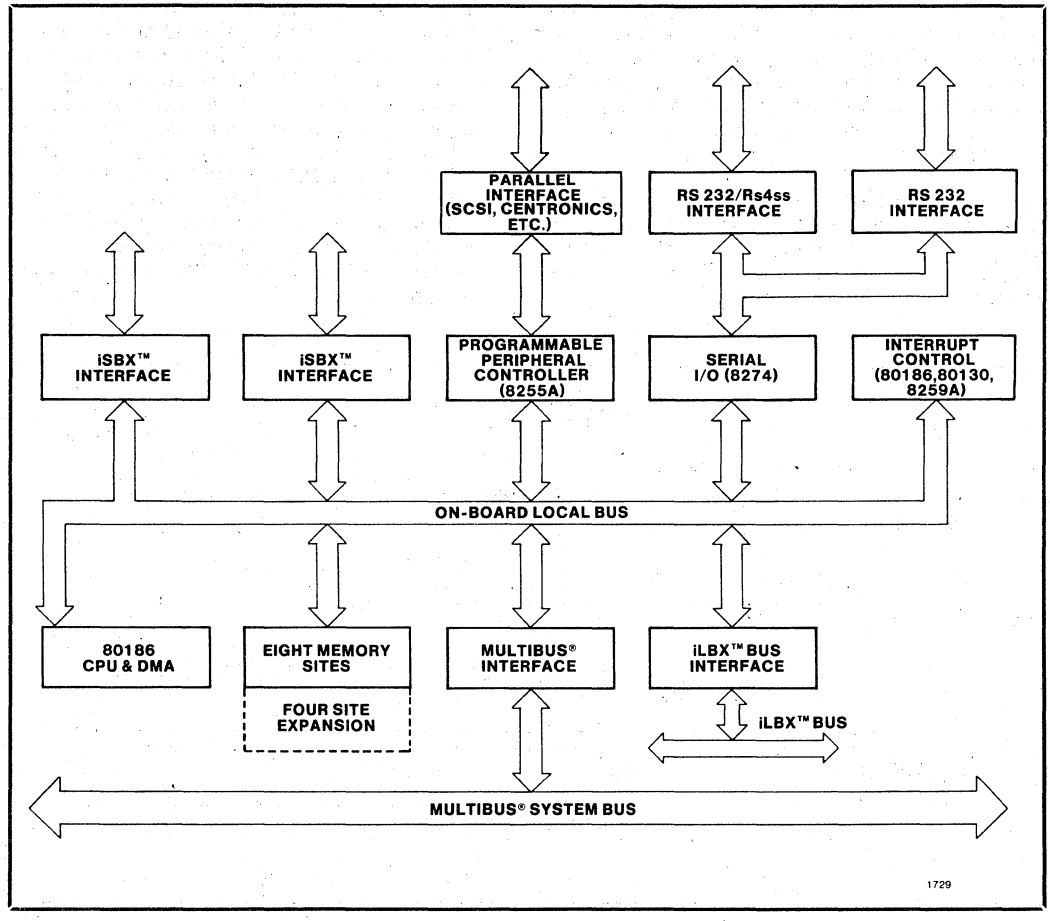


Figure 4. iSBC® 186/03 Board Block Diagram

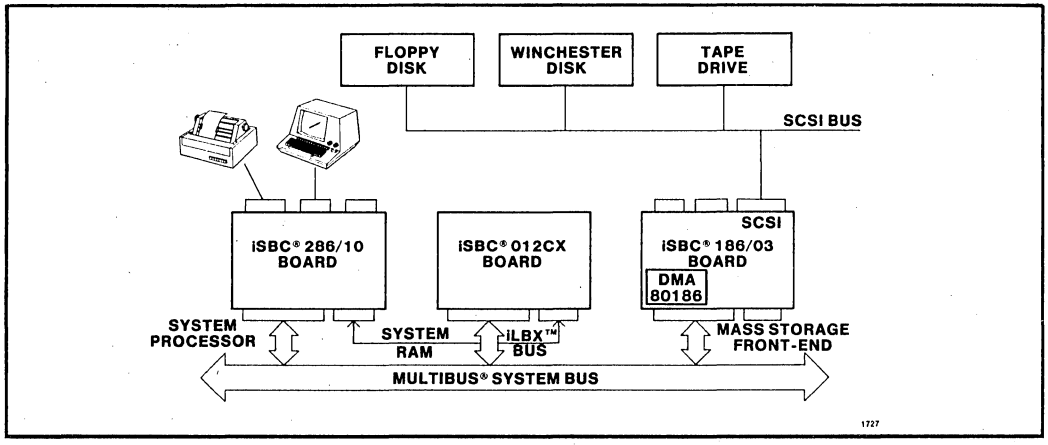


Figure 2. A Mass Storage Front-end Processor Application

STAND-ALONE SINGLE BOARD SYSTEM

A stand-alone single board system is a complete computer system on one board. By reducing the system's board count, the single board system saves space, power, and ultimately, costs. The on-board resources need to be capable of performing all of the basic system functions. These applications typically require terminal support, peripheral control, local RAM and program execution. In previous generations of Single Board Computers, these functions could only be obtained with multiple board solutions.

The iSBC 186/03 board integrates all the functions of a general purpose system (CPU, memory, I/O and peripheral control) onto one board. The iSBC 186/03 board can also be customized as a single board system by the selection of memory and iSBX™ I/O options. The board's 8 JEDEC 28-pin sockets can accommodate a wide variety of byte-wide memory devices. For example, four 27256 EPROMS and four 2186 IRAMs can be installed for a total of 128 Kb of EPROM program storage and 32 Kb of RAM data storage. If more memory is needed, an optional iSBC 341 memory site expansion board can be added to provide an additional four JEDEC sites. Two iSBX MULTIMODULE™ boards can be added to the iSBC 186/03 board to customize the board's I/O capabilities. As shown in Figure 3, the iSBX connectors can support a single-board system with the analog input and output modules needed by machine or process control systems.

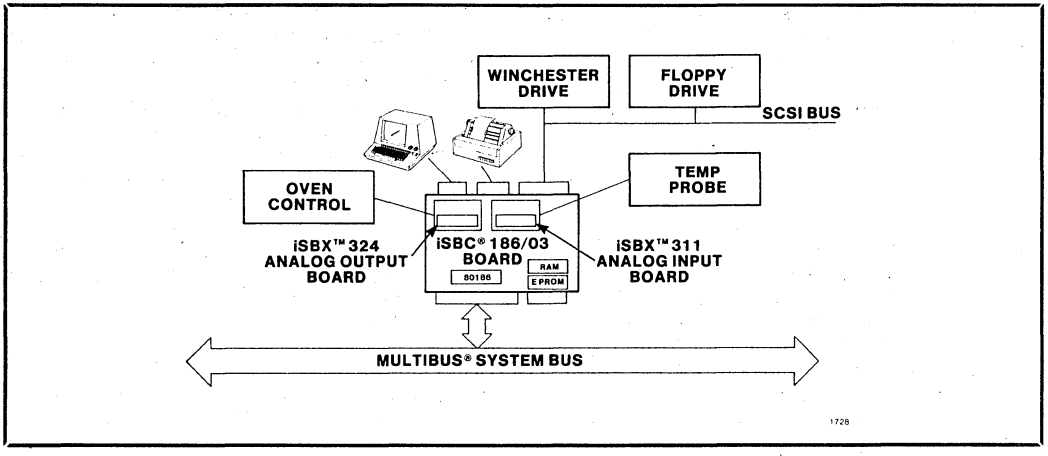


Figure 3. A Stand-alone Single Board System Application

the applications programmer, the 80130 extends the 80186 instruction set by providing 35 operating system primitive functions, and supporting five new system data types. This extension of data types and system functions makes the 80130 a logical and easy-to-use architectural extension to the 80186 system. The chip has also been designed to provide the kernal in applications using the full iRMX 86 operating system. The 80130 component may be replaced by the 80150 CP/M[1] Operating System firmware component or the 87100 user programmable firmware component.

TIMERS

The 80186 provides three internal 16-bit programmable timers. Two of these are highly flexible and are connected to four external pins (two per timer). They can be used to count external events, time external events, generate nonrepetitive waveforms, etc. As shipped on the iSBC 186/03 board, these two timers are connected to the serial interface, and provide baud rate generation. The third timer is not connected to any external pins, and is useful for real-time coding and time-delay applications. In addition, this third timer can be used as a prescaler to the other two, or as a DMA request source. The 80130 provides three more programmable timers. One is a factory default baud rate generator and outputs an 8254 compatible square wave that can be used as an alternate baud rate source to either serial channel. The 80130's second timer is used as a system timer. The third timer is reserved for use by the 80130 iRMX nucleus. The system software configures each timer independently to select the desired function. Available functions include: interrupt on terminal count, programmable one-shot, rate generator, square-wave generator, software triggered strobe, hardware triggered strobe and event counter. The contents of each counter may be read at any time during system operation.

MEMORY

There are eight JEDEC 28-pin memory sites on the iSBC 186/03 board providing flexible memory expansion. Four of these sites (EPROM sites) may be used for EPROM or E^2PROM program storage, while the other four (RAM sites) may be used for static RAM or iRAM data storage or used as additional program storage. The eight sites can be extended to twelve by the addition of an iSBC 341 JEDEC site expansion module. These additional sites may contain static RAM or iRAM devices. The EPROM sites are compatible with 8K x 8, 16K x 8 or 32K x 8 EPROMs or 2K x 8 (2817A) E^2PROMs. Device type and size are jumper selected.

Memory addressing for the JEDEC sites depends on the device type selected. The four EPROM sites are top justified in the 1 Mb address space and must contain the power-on instructions. The device size determines the starting address of these devices. The four RAM sites are, by default, located starting at address 0. The addressing of these sites may be relocated to upper memory (immediately below the EPROM site addresses) in applications where these sites will contain additional program storage. The optional iSBC 341 MULTIMODULE sites are addressable immediately above the RAM site addresses.

Power-fail control and auxiliary power are provided for protection of the RAM sites when used with static RAM devices. A memory protect signal is provided through an auxiliary connector (J4) which, when asserted, disables read/write access to RAM memory on the board. This input is provided for the protection of RAM contents during system power-down sequences. An auxiliary power bus is also provided to allow separate power to RAM for systems requiring battery back-up of read/write memory. Selection of this auxiliary RAM power bus is made via jumpers on the board.

SCSI PERIPHERAL INTERFACE

The iSBC 186/03 board includes a parallel peripheral interface that consists of three 8-bit parallel ports. As shipped, these ports are configured for general purpose I/O. The parallel interface may be reconfigured to be compatible with the SCSI disk interface by adding two user-supplied and programmed Programmable Array Logic (PAL) devices, moving jumpers and installing a user-supplied 74LS640-1 device. Alternatively, the parallel interface may be reconfigured as a DMA controlled Centronics compatible line printer interface by adding one PAL and changing jumpers.

The SCSI (Small Computer System Interconnect) interface allows multiple mass storage peripherals such as Winchester disk drives, floppy disk drives and tape drives to be connected directly to the iSBC 186/03 board. A sample SCSI application is shown in Figure 5. Intel tested iSBC 186/03 compatible SCSI controllers include Adaptek 4500, DTC 1410, Iomega Alpha 10, Shugart 1601 and 1610, Vermont Research 8103 and Xebec 1410.

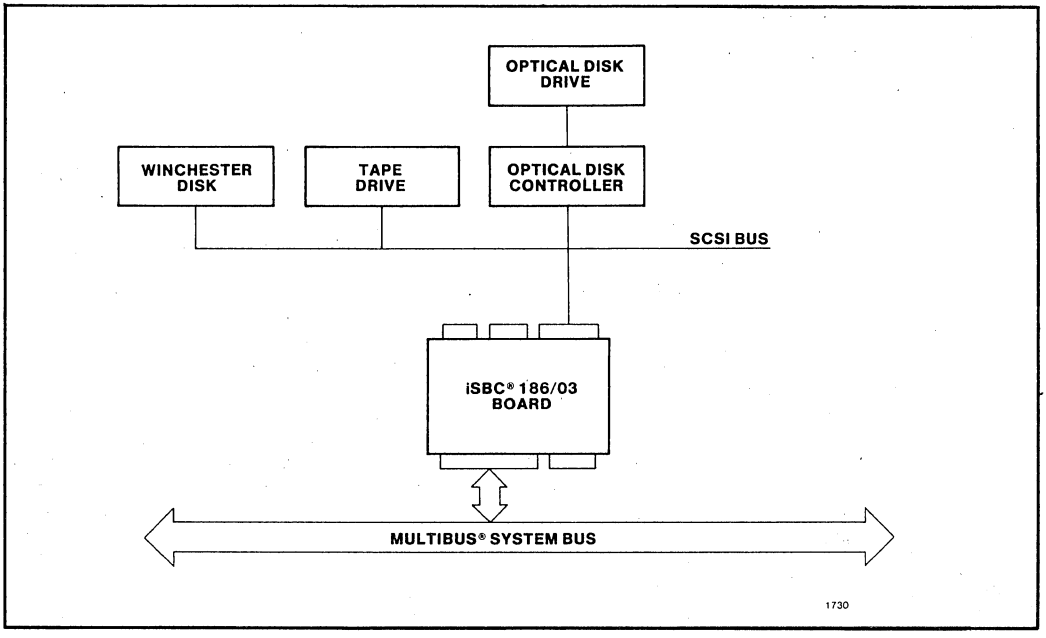


Figure 5. Sample SCSI Application

The Centronics interface requires very little software overhead since a PAL device is used to provide neccesary handshake timing. Interrupts are generated for printer fault conditions and a DMA request is issued for every character. The interface supports Centronics type printers compatible with models 702 or 737.

SERIAL I/O

The iSBC 186/03 single board computer contains two programmable communications interfaces using the Intel 8274 Multi-Protocol Serial Controller (MPSC). Two 80186 timer outputs are used as software selectable baud rate generators capable of supplying the serial channels with common communications frequencies. An 80130 baud rate timer may be jumpered to either serial port to provide higher frequency baud rates. The mode of operation (i.e., Asynchronous, Byte Synchronous or Bisynchronous protocols), data format, control character format, parity, and baud rate are all under program control. The 8274 provides full duplex, double buffered transmit and receive capability. Parity, overrun, and framing error detection are all incorporated in the MPSC. The iSBC 186/03 board supports operation in the polled, interrupt and DMA driven interfaces though jumper options. The default configuration is with channel A as RS422A/RS449, channel B as RS232C. Channel A can optionally be configured to support RS232C. Both channels are default configured as data set (DCE). Channel A can be reconfigured as data terminal (DTE) for connection to a modem-type device.

INTERRUPT CONTROL

The iSBC 186/03 board provides 27 on-board vectored interrupt levels to service interrupts generated from 33 possible sources.

The interrupts are serviced by four programmable interrupt controllers (PICs): one in the 80186 component, one in the 80130 component, one in the 8259A component and one in the 8274 component. The 80186, 8259A and 8274 PICs act as slaves to the 80130 master PIC. The highest priority interrupt is the Non-Maskable Interrupt (NMI) line which is tied directly to the 80186 CPU. This interrupt is typically used to signal catastrophic events (e.g. power failure). The PICs provide prioritization and vectoring for the other 26 interrupt requests from on-board I/O resouces and from the MULTIBUS system bus. The PICs then resolve the requests according to the programmable priority resolution mode, and if appropriate, issue an interrupt to the CPU.

Table 1 contains a list of devices and functions capable of generating interrupts. These interrupt sources are jumper configurable to the desired interrupt request level.

EXPANSION

OVERVIEW

The iSBC 186/03 board architecture includes three bus structures: the MULTIBUS system bus, the iLBX local bus expansion and the iSBX MULTIMODULE expansion bus as shown in Figure 6. Each bus structure is optimized to satisfy particular system requirements. The system bus provides a basis for general system design including memory and I/O expansion as well as multiprocessing support. The iLBX bus allows large amounts of high performance memory to be accessed by the iSBC 186/03 board over a private bus. The iSBX MULTIMODULE expansion board bus is a means of adding inexpensive I/O functions to the iSBC 186/03 board. Each of these three bus structures are implemented on the iSBC 186/03 board providing a flexible system architecture solution.

MULTIBUS® SYSTEM BUS - IEEE 796

The MULTIBUS system bus is an industry standard (IEEE 796) microcomputer bus structure. Both 8- and 16-bit single board computers are supported on the IEEE 796 structure with 20 or 24 address and 16 data lines. In its simplest application, the system bus allows expansion of functions already contained on a single board computer (e.g., memory and I/O). However, the IEEE 796 bus also allows very powerful distributed processing configurations with multiple processors and intelligent slave, I/O and peripheral boards capable of solving the most demanding microcomputer applications. The MULTIBUS system bus is supported with a broad array of board-level products, LSI interface components, detailed published specifications and application notes.

Table 1. Interrupt Request Sources

DEVICE	FUNCTION	NUMBER OF INTERRUPTS
MULTIBUS interface INTO - INT7	Requests from MULTIBUS resident peripherals or other CPU	8
8274 Serial Controller	Transmit buffer empty, receive buffer full and channel errors	8
Internal 80186 Timer and DMA	Timer 0,1,2, outputs (function determined by timer mode) and 2 DMA channel interrupts	5
80130 Timer Output	RMX system timer (SYSTICK)	1
iSBX connectors	Function determined by iSBX MULTIMODULE board	6 (3 per iSBX connector)
Bus fail-safe timer	Indicates addressed MULTIBUS resident device has not responded to command within 10 msec	1
8255A Parallel I/O Controller	Parallel port control	2
J4 Connector	External/Power-fail interrupts	2

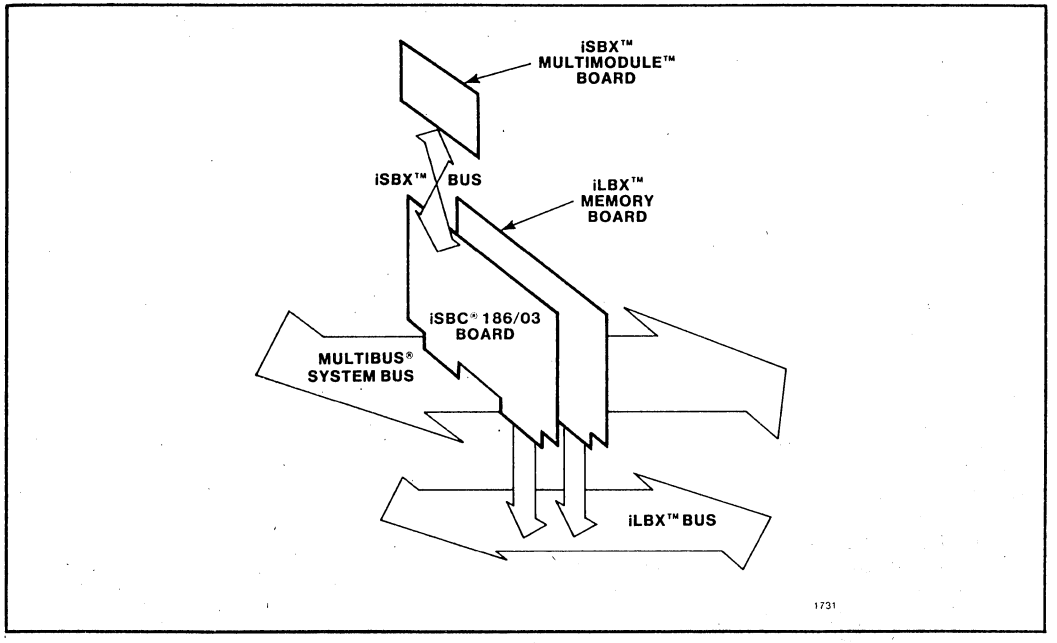


Figure 6. iSBC® 186/03 Board System Architecture

iLBX™ BUS - LOCAL BUS EXTENSION

The iSBC 186/03 board provides a local bus extension (iLBX) interface. This standard extension allows on-board memory performance with physically off-board memory. The combination of a CPU board and iLBX memory boards is architecturally equivalent to a single board computer and thus can be called a "virtual Single Board Computer". The iLBX bus is implemented over the P2 connector and requires independent cabling or backplane connection

iSBX™ BUS MULTIMODULE™ ON-BOARD EXPANSION

Two iSBX MULTIMODULE connectors are provided on the iSBC 186/03 microcomputer board. Through these connectors, additional on-board I/O functions may be added. iSBX MULTIMODULE boards optimally support functions provided by VLSI peripheral components such as additional parallel and serial I/O, analog I/O, and graphics control. The iSBX connectors on the iSBC 186/03 board provide all signals necessary to interface to the local on-board bus, including 16 data lines for maximum data transfer rates. MULTIMODULE boards designed with 8-bit data paths and using the 8-bit iSBX connector are also supported on the iSBC 186/03 microcomputer board. A broad range of iSBX MULTIMODULE options are available from Intel. Custom iSBX modules may also be designed. An iSBX bus interface specification and iSBX connectors are available from Intel.

OPERATING SYSTEM SUPPORT

Intel's iRMX 86 Operating System is a highly functional operating system with a very rich set of features and options based on an object-oriented architecture. In addition to being modular and configurable, functions include a sophisticated file management and I/O system, and a powerful human interface. The iRMX 86 release 6 operating system can be used with the iSBC 186/03 board to generate application code for iRMX 86 based systems.

Intel also offers an iOSP™ 86 Operating System Firmware package that provides the initialization code and configuration utilities required to use the iSBC 186/03 board's 80130 iRMX 86 nucleus firmware. This package enables the 80130 code to be used as is or to be extended with the other features of the iRMX 86 release 6 operating system.

The 80130 iRMX 86 firmware device may be replaced with a user supplied 80150 CP/M firmware device. Using this device, the iSBC 186/03 board can provide a total CP/M 86 system on a single board.

DEVELOPMENT ENVIRONMENT

Intel offers numerous tools to aid in the development of iSBC 186/03 applications. These include on-target development, full development systems, in-circuit emulators and programming languages. Some of the features of each are described below.

Using the iRMX 86 release 6 operating system, software development can be performed directly on the iSBC 186/03 board. This on-target development is the most economical way to develop iSBC 186/03 based projects.

The development cycle of iSBC 186/03 products can be significantly reduced and simplified by using either the System 86/3XX (iRMX 86 based) or the Intellec® Series Microcomputer Development Systems. The Assembler, Locater Linker, Library Manager, Text Editor and System Monitor are all supported by the ISIS-II disk-based operating system. To facilitate conversion of 8080A/8085A assembly language programs to run on the iSBC 186/03 boards, CONV-86 is available under the ISIS-II operating system.

The Integrated Instrumentation In-Circuit Emulator (I^2ICE^{TM}) provides the necessary link between an Intellec development system and the "target" iSBC 186/03 execution system. In addition to providing the mechanism for loading executable code and data into the iSBC 186/03 boards, the I^2ICE 186 emulator provides a sophisticated command set to assist in debugging software and final integration of the user hardware and software.

Intel has two systems implementation languages, PL/M-86 and C-86. Both are available for use on the iRMX 86 operating system, on the System 86/3XX and on the Intellec Microcomputer Development System. PL/M-86 provides the capability to program in algorithmic language and eliminates the need to manage register usage or allocate memory while still allowing explicit control of the system's resources when needed. C-86 is especially appropriate in applications requiring portability and code density. FORTRAN 86, PASCAL 86, and BASIC 86 are also available on the iRMX 86 operating system, on the System 86/3XX and on the Intellec development system.

SPECIFICATIONS

Word Size

Instruction — 8, 16, 24 or 32 bits
Data — 8 or 16 bits

System Clock

6.0 Mhz

Basic Instruction Cycle Time

1 μs
333 ns (assumes instruction in the queue)

Note: Basic instruction cycle is defined as the fastest instruction time (i.e. two clock cycles plus instruction fetch). Zero wait-state memory is assumed.

Memory Response Times

	Max. Access Time (from chip enable)	Min. Cycle Time
EPROM Memory Sites		
0 wait states	347ns	423ns
1 wait state	513ns	589ns
RAM Memory Sites		
with SRAMs or EPROMS		
0 wait states	302ns	423ns
1 wait state	468ns	589ns
with 2186 IRAMs		
1 wait state	336ns	589ns
2 wait states	560ns	755ns

Note: The number of wait states inserted is jumper selected depending on memory device specifications.

Memory Capacity/Addressing

Four EPROM Sites

Device	Capacity	Address Range
2764 EPROM	32Kb	$F8000_H$ - $FFFFF_H$
27128 EPROM	64Kb	$F0000_H$ - $FFFFF_H$
27256 EPROM	128Kb	$E0000_H$ - $FFFFF_H$

Four RAM Sites

Device	Capacity	Address Range
2K SRAM	8Kb	0 - $01FFF_H$
8K SRAM	32Kb	0 - $07FFF_H$
2186 RAM	32Kb	0 - $07FFF_H$
2817A E^2PROM	8K (must use 8K x 8 decode option, there are four copies of the E^2PROM in the 8K x 8 address area)	$F0000_H$ - $F7FFF_H$
2764 EPROM	32Kb	$F0000_H$ - $F7FFF_H$ (below EPROM sites)
27128 EPROM	64Kb	$E0000_H$ - $EFFFF_H$ (below EPROM sites)
27256 EPROM	128Kb	$C0000_H$ - $DFFFF_H$ (below EPROM sites)

Four iSBC 341 Expansion Sites

Device	Capacity	Address Range
2K SRAM	8Kb	02000_H - $03FFF_H$
8K SRAM	32Kb	08000_H - $0FFFF_H$
2186 RAM	32Kb	08000_H - $0FFFF_H$
2817A E^2PROM	8Kb (may be mixed with 2K x 8 SRAM)	02000_H - $03FFF_H$

Note: All on-board memory is local to the CPU (ie. not dual-ported)

Serial Communications Characteristics

Synchronous — 5-8 bit characters, internal or external character synchronization; automatic sync insertion; break character generation

Asynchronous — 5-8 bit characters; 1, ½, or 2 stop bit; false start bit detection.

Common Baud Rates

Using 80186 timers:	Using 80130 timer:
	500K
125K	125K
64K	64K
48K	48K
19.2K	19.2K
9600	9600
4800	4800
2400	2400
1200	1200
600	600
300	300
150	150
110*	110*
75*	75*

* Asynchronous use only

Note: Frequency selected by I/O write of appropriate 16-bit frequency factor to baud rate register of 80186 or 80130 timers.

Timer Input Frequency

80186 Reference: 1.5 MHz ± 0.1%

80130 Reference: 6.0 MHz ± 0.1%

Interface Compliance

MULTIBUS - IEEE 796 compliance: Master D16 M24 I16 VO EL

iSBX Bus - Two 8/16 bit iSBX connectors allow use of up to 2 single-wide modules or 1 single-wide and 1 double-wide module. Intel iSBX bus compliance: D16/16 DMA

iLBX Bus - Intel iLBX bus compliance: PM D16

Serial I/O - Channel A: Configurable as RS 422A or RS 232C compatible, configurable as a data set or data terminal
Channel B: RS 232C compatable, configured as data set

Parallel I/O - SCSI (ANSI - X3T9, 2/82-s) compatible or Centronics 702 or 737 compatible (requires user supplied PALs and 74LS640-1)

CONNECTORS

Interface	Double-sided Pins	Mating Connectors
MULTIBUS System	86 (P1)	Viking 3KH43/9AMK12 Wire Wrap
	60 (P2)	Viking 3KH30/9JNK
iSBX Bus 8-Bit Data	36	Viking 000292-0001
16-Bit Data	44	Viking 000293-0001
Serial I/O	26	3M 3452-0001 Flat AMP88106-1 Flat
iLBX Bus	60	Kelam RF30-2853-542
Parallel Interface	50	3M 3425-6000 3M 3425-6050 w/strain Ansley 609-5001M

PHYSICAL CHARACTERISTICS

Width — 12.00 in. (30.48 cm)
Length — 7.05 (17.90 cm)
Height — 0.50 in. (1.78 cm)
Weight — 13 ounces

ENVIRONMENTAL CHARACTERISTICS

Operating Temperature — 0° C to 55 ° C at 200 Linear Feet/Min (LFM) Air Velocity
Relative Humidity — 90% (without condensation)

ELECTRICAL CHARACTERISTICS

The maximum power required per voltage is shown below. These numbers do not include the power required by the optional memory devices, SCSI PALs, battery back-up or expansion modules.

Voltage (volts)	Max. Current (amps)	Max Power (watts)
+ 5	4.8	24
+ 12	.04	.48
− 12	.04	.48

REFERENCE MANUAL

Guide to Using the iSBC® 186/03 Single Board Computer — Order Number 145921

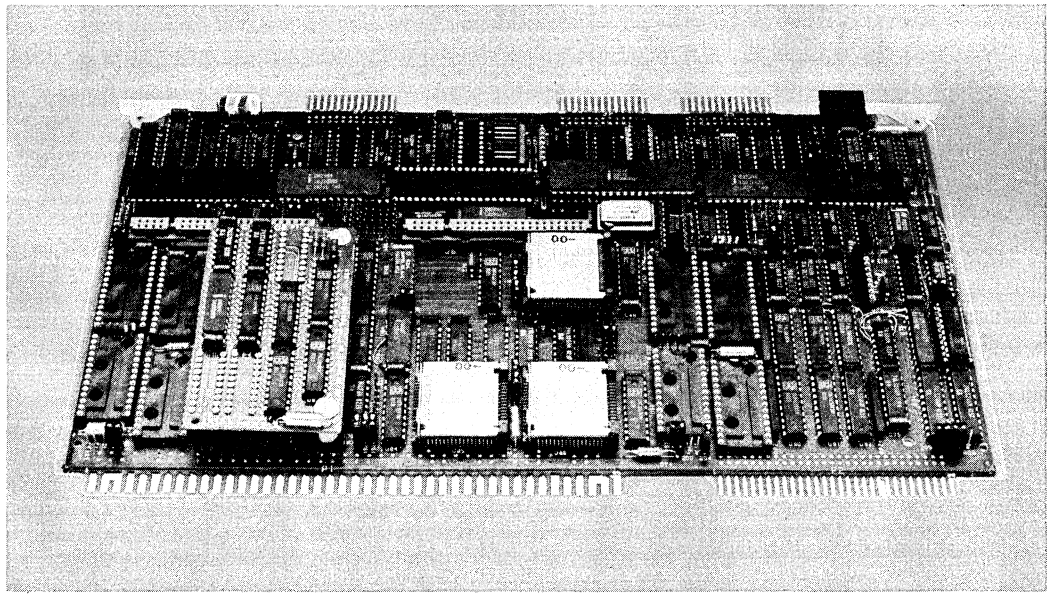
ORDERING INFORMATION

Part Number	Description
SBC 186/03	186-based single board computer

iSBC® 286/10 SINGLE BOARD COMPUTER

- iAPX 286/10 (80286) Microprocessor with 6.0 MHz CPU clock
- Optional 80287 Numeric Data Processor
- iLBX™ (Local Bus Extension) interface for high-speed memory expansion
- Two iSBX™ bus interface connectors for I/O expansion
- Eight JEDEC 28-pin sites for optional RAM/iRAM/EPROM/E^{2}PROM components
- Optional expansion to twelve JEDEC 28-pin sites with an iSBC® 341 28-pin site expansion board
- 16 levels of vectored interrupt control
- Centronics-compatible parallel I/O printer interface
- Two programmable multiprotocol synchronous/asynchronous serial interfaces; one RS232C, the other RS232C or RS422 compatible
- MULTIBUS® interface for multimaster configurations and system expansion

The iSBC® 286/10 Single Board Computer is a member of Intel's complete line of microcomputer modules and systems which take advantage of Intel's VLSI technology to provide economical, off-the-shelf, computer-based solutions for OEM applications. The board is a complete microcomputer system on a 6.75 x 12.0 inch printed circuit card. The CPU, system clock, memory sockets, I/O ports and drivers, serial communications interface, priority interrupt logic and programmable timers, all reside on the board. The iSBC 286/10 board is the first single board computer to incorporate the iAPX 286 CPU and the iLBX™ bus extension. This combination provides the highest performance 16-bit microcomputer system solution. The iLBX architectural expansion maintains this high performance for applications requiring vast amounts of system memory.



ORDER NO: 280079-002

FUNCTIONAL DESCRIPTION

Overview

The iSBC 286/10 board utilizes the powerful iAPX 286 CPU within the MULTIBUS system architecture, enhanced by the iLBX bus, to provide a high performance 16-bit solution. This board also includes on-board interrupt, memory and I/O features facilitating a complete single board computer system.

Central Processing Unit

The central processor for the iSBC 286/10 board is the 80286 CPU operating at a 6.0 MHz clock rate. The 80286 CPU is upwardly compatible with Intel's iAPX 88 and iAPX 86 CPUs. The 80286 CPU runs iAPX 88 and 86 code at substantially higher speeds due to a parallel chip architecture. In addition, the 80286 CPU provides on chip memory management and protection and virtual memory addressing of up to 1 gigabyte per task. Numeric processing power may be enhanced with the optional 80287 numerics processor. The clock rates for the 80286 and the 80287 are independent with the 80287 rate jumper selectable at either 4.0 or 8.0 MHz.

Instruction Set

The 80286 instruction repertoire includes variable length instruction format (including double operand instructions), 8-bit and 16-bit signed and unsigned arithmetic operators for binary, BCD and unpacked ASCII data, and iterative word and byte string manipulation functions.

For enhanced numerics processing capability, the 80287 Numeric Data Processor extends the 80286 architecture and data set. Over 60 numeric instructions offer arithmetic, trigonometric, transcendental, logarithmic and exponential instructions. Supported data types include 16-, 32-, and 64-bit integer, 32- and 64-bit floating point, 18-digit packed BCD and 80-bit temporary. The 80287 meets the proposed IEEE P754 standard for numeric data processing and maintains compatibility with 8087-based systems.

Architectural Features

The iAPX 86, 88, 186, and 286 CPU family all contain the same basic set of registers, instructions, and addressing modes. The 80286 processor is upward compatible with the 8086, 8088 and 80186 CPUs.

The 80286 operates in two modes: iAPX 86 real address mode and protected virtual address mode. In iAPX 86

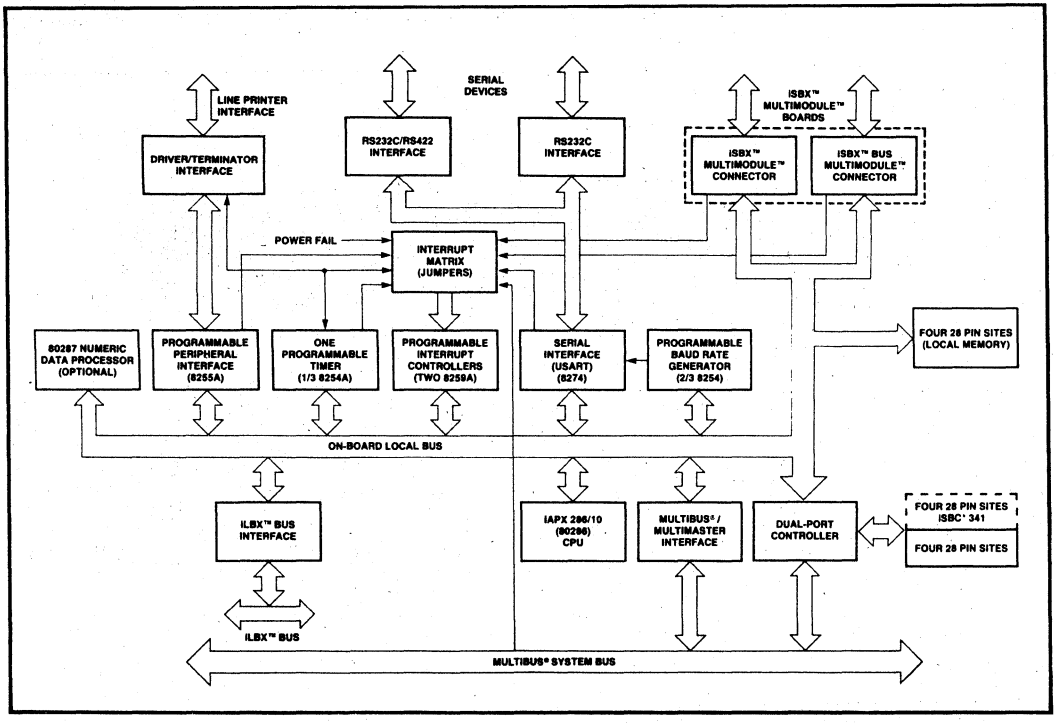


Figure 1. iSBC® 286/10 Block Diagram

280079-002

real address mode, programs use real addresses with up to one megabyte of address space. Programs use virtual addresses in protected virtual address mode, also called protected mode. In protected mode, the 80286 CPU automatically maps 1 gigabyte of virtual addresses per task into a 16 megabyte real address space. This mode also provides memory protection to isolate the operating system and ensure privacy of each tasks' programs and data. Both modes provide the same base instruction set, registers, and addressing modes.

VECTORED INTERRUPT CONTROL

Incoming interrupts are handled by two on-board 8259A programmable interrupt controllers and by the 80286's NMI line. Interrupts originating from up to 16 sources are prioritized and then sent to the CPU as a vector address. Further interrupt capability is available through bus vectored interrupts where slave 8259 interrupt controllers resident on separate SBC boards and are cascaded into the on-board interrupt control.

INTERRUPT SOURCES

Twenty-three potential interrupt sources are routed to the interrupt jumper matrix where the user can connect the desired interrupt sources to specific interrupt levels. Table 1 includes a list of devices and functions supported by interrupts.

MEMORY CAPABILITIES

There are eight 28-pin JEDEC sites on-board which may contain a combination of byte-wide devices including RAM, iRAM, EPROM, and E^2PROM. These sites are organized into two 4-site blocks, one of which may be dual-ported. The dual port block may be extended to eight sites (i.e. 12 sites total) by the addition of an iSBC 341 JEDEC site expansion module. The on-board EPROM capacity using twelve 27128 EPROMs is 192 Kbytes. The on-board RAM using ten 8K x 8 RAMs is 80 Kbytes.

SERIAL I/O

A two channel serial communications interface using Intel's 8274 Multi-Protocol Serial Controller (MPSC) is contained on the iSBC 286/10 board. Two independent software selectable baud rate generators provide the MPSC with all common communication frequencies. The protocol (i.e. asynchronous, IBM bisync, or SDLC/HDLC), data format, control character format, parity and baud rate are all under program control. Software interfacing to the MPSC can be via either a polled or interrupt driven routine. One channel may be configured for an RS232C or RS422 interface with the other channel RS232C only. The data, command and signal ground lines for each channel are brought out to two 26-pin connectors.

Table 1. Interrupt Request Sources

Device	Function	Number of Interrupts
MULTIBUS® interface	Requests from MULTIBUS® resident peripherals or other CPU boards	8*
8259A programmable interrupt controller	8 level vectored interrupt request cascaded to master 8259A	1
8274 serial controller	8 level vectored interrupt request cascaded to master 8259A	1
8255A line printer interface	Signals output buffer empty	1
8254 timers	Timer 0, 1 outputs; function determined by timer mode	2
iSBX™ connectors	Function determined by iSBX™ MULTIMODULE™ board	4 (2 per iSBX™ connector)
Bus fail safe timer	Indicates addressed MULTIBUS® resident device has not responded to command within 6 msec	1
Power fail interrupt	Indicates AC power is not within tolerance	1
External interrupt	General purpose interrupt from auxiliary connector, commonly used as front panel interrupt	1
On-board logic	Conditioned interrupt source from edge sense latch, inverter, or OR gate	3

* May be expanded to 56 with slave 8259A PICs on MULTIBUS® boards

280079-002

PROGRAMMABLE TIMERS

The iSBC 286/10 board provides three independent, fully programmable 16-bit interval timers/event counters utilizing the Intel 8254 Programmable Interval Timer. Each counter is capable of operating in either BCD or binary modes. Two of these timers/counters are available to the systems designer to generate accurate time intervals under software control. Routing for the outputs of these counters is jumper selectable. The outputs may be independently routed to the 8259A Programmable Interrupt Controller or to the 8274 MPSC to count external events or provide baud rate generation. The third interval timer in the 8254 is dedicated to providing a clock for the programmable baud rate generator in the iSBC 286/10 board's MPSC serial controller. The system software configures each timer independently to select the desired function. Seven functions are available as shown in Table 2. The contents of each counter may be read at any time during system operation.

Table 2. Programmable Timer Functions

Function	Operation
Interrupt on terminal count	When terminal count is reached, an interrupt request is generated. This function is extremely useful for generation of real-time clocks.
Programmable one-shot	Output goes low upon receipt of an external trigger edge or software command and returns high when terminal count is reached. This function is retriggerable.
Rate generator	Divide by N counter. The output will go low for one input clock cycle, and the period from one low going pulse to the next is N times the input clock period.
Square-wave rate generator	Output will remain high until one-half the count has been completed, and go low for the other half of the count.
Software triggered strobe	Output remains high until software loads count (N). N counts after count is loaded, output goes low for one input clock period.
Hardware triggered strobe	Output goes low for one clock period N counts after rising edge counter trigger input. The counter is retriggerable.
Event counter	On a jumper selectable basis, the clock input becomes an input from the external system. CPU may read the number of events occurring after the counter "window" has been enabled or an interrupt may be generated after N events occur in the system.

LINE PRINTER INTERFACE

An 8255A Programmable Peripheral Interface (PPI) provides a line printer interface, several on-board functions, and four non-dedicated input bits. Drivers are provided for a complete Centronics compatible line printer interface. The on-board functions implemented with the PPI are power fail sense, override, NMI mask, non-volatile RAM enable, clear timeout interrupt, LED 0 and 1, clear edge sense flop, MULTIBUS interrupt, and serial channel A loopback. The PPI's I/O lines are divided into three eight bit ports: A, B and C. Four non-dedicated input bits allow the state of four user configured jumper connections to be input. The PPI must be programmed for mode 0 with ports A and C used as outputs and port B as input. A "dummy" write to port B is used to set the iSBC 286/10 board to protected mode. The parallel port bit assignment is shown in Table 3.

Table 3. Parallel Port Bit Assignment

Port A — Output	
Bit	**Function**
0	Line Printer Data Bit 0
1	Line Printer Data Bit 1
2	Line Printer Data Bit 2
3	Line Printer Data Bit 3
4	Line Printer Data Bit 4
5	Line Printer Data Bit 5
6	Line Printer Data Bit 6
7	Line Printer Data Bit 7
Port B — Input	
Bit	**Function**
0	General Purpose Input 0
1	General Purpose Input 1
2	General Purpose Input 2
3	General Purpose Input 3
4	Line Printer ACK/ (Active Low)
5	Power Fail Sense/ (Active Low)
6	Line Printer Error (Active Hi)
7	Line Printer Busy (Active Hi)
Port C — Output	
Bit	**Function**
0	Line Printer Data Strobe (Active Hi)
1	Override/ (Active Low)
2	NMI Mask (0 = NMI Enabled)
3	Non-Volatile RAM Enable; Clear Timeout Interrupt/
4	LED 0 (1 = On); Clear Edge Sense Flop/
5	MULTIBUS® Interrupt (1 = Active)
6	Serial CHA Loopback (0 = Online, 1 = Loopback)
7	LED 1 (1 = On); Clear Line Printer Ack Flop/

280079-002

MULTIBUS® System Architecture

OVERVIEW

The MULTIBUS system architecture includes three bus structures: the system bus, the local bus extension and the MULTIMODULE™ expansion bus as shown in Figure 2. Each bus structure is optimized to satisfy particular system requirements. The system bus provides a basis for general system design including memory and I/O expansion as well as multiprocessing support. The local bus extension alllows large amounts of high performance memory to be accessed from a CPU board over a private bus. The MULTIMODULE extension bus is a means of adding inexpensive I/O functions to a base CPU board. Each of these three bus structures are implemented on the iSBC 286/10 board providing a total system architecture solution.

SYSTEM BUS – IEEE 796

The MULTIBUS system bus is Intel's industry standard, IEEE 796, microcomputer bus structure. Both 8- and 16-bit single board computers are supported on the IEEE 796 structure with 24 address and 16 data lines. In its simplest application, the system bus allows expansion of functions already contained on a single board computer (e.g., memory and digital I/O). However, the IEEE 796 bus also allows very powerful distributed processing configurations with multiple processors and intelligent slave, I/O and peripheral boards capable of solving the most demanding microcomputer applications. The MULTIBUS system bus is supported with a broad array of board level products, LSI interface components, detailed published specifications and application notes.

SYSTEM BUS – EXPANSION CAPABILITIES

Memory and I/O capacity may be expanded and additional functions added using Intel MULTIBUS compatible expansion boards. Memory may be expanded by adding user specified combinations of RAM boards, EPROM boards, or combination boards. Input/output capacity may be added with digital I/O and analog I/O expansion boards. Mass storage capability may be achieved by adding single or double density diskette controllers, or hard disk controllers. Modular expandable backplanes and cardcages are available to support multiboard systems.

SYSTEM BUS – MULTIMASTER CAPABILITIES

For those applications requiring additional processing capacity and the benefits of multiprocessing (i.e., several CPUs and/or controllers logically sharing system tasks through communication of the system bus), the iSBC 286/10 board provides full system bus arbitration control logic. This control logic allows up to three iSBC

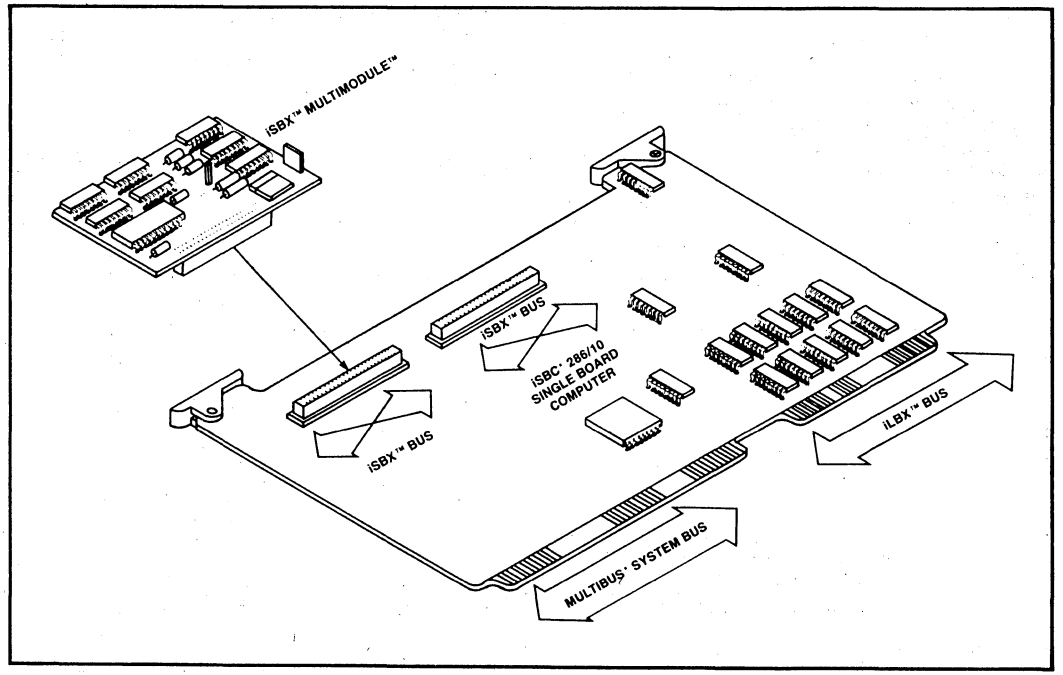


Figure 2. MULTIBUS® System Architecture

280079-002

286/10 boards or other bus masters, including iSBC 80 family MULTIBUS compatible 8-bit single board computers to share the system bus using a serial (daisy chain) priority scheme and allows up to 16 masters to share the MULTIBUS system bus with an external parallel priority decoder. In addition to multiprocessing configuration made possible with multimaster capability, it also provides a very efficient mechanism for all forms of DMA (Direct Memory Access) transfers.

iLBX™ BUS – LOCAL BUS EXTENSION

The iSBC 286/10 board also provides the local bus extension (iLBX) of the MULTIBUS architecture. This standard extension allows on-board memory performance with physically off-board memory. The combination of a CPU board and iLBX memory boards is architecturally equivalent to a single board computer and thus can be called a "virtual SBC". The iLBX is implemented over the P2 connector and requires cabling across the virtual SBCs of a system (see Figure 3). Other Intel products which support the iLBX bus include:

iSBC 028CX 128KB iLBX RAM board
iSBC 056CX 256KB iLBX RAM board
iSBC 012CX 512KB iLBX RAM board
iSBC 428 JEDEC 28-PIN SITE board
iSBC 580 MULTICHANNEL™ interface board

iSBX™ BUS MULTIMODULE™ ON-BOARD EXPANSION

Two 8/16-bit iSBX™ MULTIMODULE connectors are provided on the iSBC 286/10 microcomputer board. Through these connectors, additional on-board I/O functions may be added. iSBX MULTIMODULEs optimally support functions provided by VLSI peripheral components such as additional parallel and serial I/O, analog I/O, small mass storage device controllers (e.g., cassettes and floppy disks), and other custom interfaces to meet specific needs. By mounting directly on the single board computer, less interface logic, less power, simpler packaging, higher performance, and lower cost result when compared to other alternatives such as MULTIBUS form factor compatible boards. The iSBX connectors on the iSBC 286/10 board provide all signals necessary to interface to the local on-board bus, including 16 data lines for maximum data transfer rates. iSBX MULTIMODULE boards designed with 8-bit data paths and using the 8-bit iSBX connector are also supported on the iSBC 286/10 microcomputer board. A broad range of iSBX MULTIMODULE options are available from Intel. Custom iSBX modules may also be designed. An iSBX bus interface specification and iSBX connectors are available from Intel.

Software Support

Real time support for the iSBC 286/10 board is provided by Release 6 of the iRMX 86 operating system. Release 6 of the iRMX 86 operating system is an adaption of the iRMX 86 nucleus to operate on the iSBC 286/10 board in real address mode. Release 6 of the iRMX 86 enhances the ICU for configuration support of the board, adds a driver for the on-board 8274 and supports the 80287. Release 6 of iRMX 86 is completely compatible with earlier versions of iRMX 86.

Intel's family of iRMX operating systems provide high-performance, real time, multitasking O.S. support for Intel's single board computers. iRMX employs a highly configurable, modular structure for easy system configuration and expansion. The iRMX family offers a wealth of design facilities and industry standard languages to support fast, easy development.

Interactive multi-user support will be provided by the XENIX[1] operating system. XENIX is a compatible derivative of UNIX[2], System III.

Language support for the iSBC 286/10 boards real address mode includes Intel's ASM 86, PL/M 86, PASCAL and FORTRAN as well as many third party 8086 languages. Language support for virtual address mode

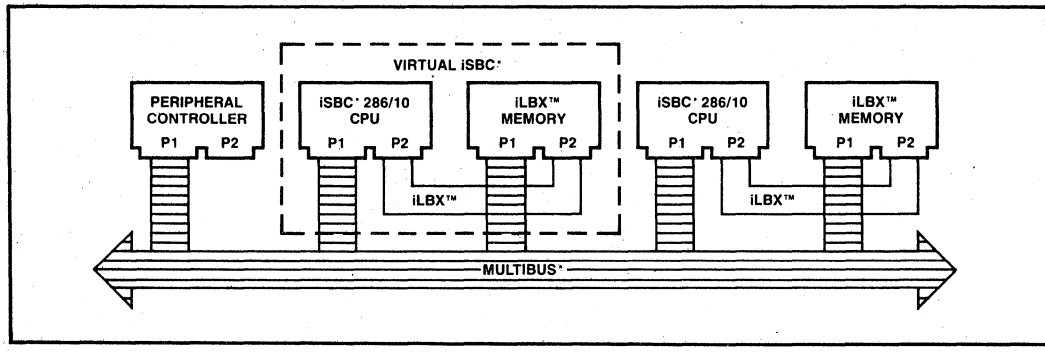


Figure 3. MULTIBUS®/iLBX™ Configuration

280079-002

operation includes ASM 286, PL/M 286, PASCAL and C. Programs developed in these languages can be downloaded from an Intel Series III Development System to the iSBC 286/10 board via the iSDM™ 286 System Debug Monitor. The iSDM 286 monitor also provides on-target program debugging support including breakpoint and memory examination features.

[1] Xenix is a trademark of Microsoft Inc.
[2] UNIX is a trademark of Bell Labs.

SPECIFICATIONS

Word Size

Instruction — 8, 16, 24, 32 or 40 bits

Data — 8 or 16 bits

System Clock

CPU — 6.0 MHz

Numeric Processor — 4.0 or 8.0 MHz (Jumper Selectable)

Cycle Time

Basic Instruction — 6.0 MHz - 500 ns; 333 ns (assumes instruction in queue)

NOTE: Basic instruction cycle is defined as the fastest instruction time (i.e. two clock cycles).

Memory Capacity (Maximum)

EPROM — 2716, 8 Kbytes; 2732, 16 Kbytes; 2764, 64 Kbytes; 27128, 128 Kbytes; 27256, 256 Kbytes

E²PROM — 2817A, 16 Kbytes

iRAM — 2186, 16 Kbytes

Static RAM — 8K x 8 devices, 48 Kbytes

NOTES: Two local sites must contain boot-up EPROM or E²PROM. 2716s and 2732s may reside in dual-port sites only. iRAMs may reside in local sites only.

WITH iSBC® 341 MULTIMODULE™

EPROM — 2716, 16 Kbytes; 2732, 32 Kbytes; 2764, 96 Kbytes; 27128, 192 Kbytes; 27256, 256 Kbytes

E²PROM — 2817A, 24 Kbytes

iRAM — 2186, 16 Kbytes

Static RAM — 8K x 8 devices, 80 Kbytes

NOTES: Dual-port sites can address 128 Kbytes of memory maximum. Two local sites must contain boot-up EPROM or E²PROM. 2716s and 2732s may reside in dual-port sites only. iRAMs may reside in local sites only.

I/O Capability

Parallel — Line printer interface, on-board fuctions, and four non-dedicated input bits

Serial — Two programmable channels using one 8274

Timers — Three programmable timers using one 8254

Expansion — Two 8/16-bit iSBX MULTIMODULE connectors

Interrupt Capacity

Potential Interrupt Sources — 23, jumper selectable

Interrupt Levels — 16 vectored requests using two 8259As and the 80286's NMI line.

Serial Communications Characteristics

Synchronous — 5-8 bit characters; internal or HDLC/SDLC character synchronization; automatic sync insertion; even or odd parity.

Asynchronous — 5-8 bit characters; break character generation; 1, 1½, or 2 stop bits; false start bit detection; even or odd parity.

280079-002

BAUD RATES

Frequency (kHz) (Software Selectable)	Baud Rate (Hz)				
	Synchronous	Asynchronous			
Reference: 1.23 MHz	÷1	÷1	÷6	÷32	÷64
615.	615,000	615,000	38,400	19,200	9,600
307.	307,000	307,000	19,200	9,600	4,800
154.	154,000	154,000	9,600	4,800	2,400
76.8	76,800	76,800	4,800	2,400	1,200
38.4	38,400	38,400	2,400	1,200	600
19.2	19,200	19,200	1,200	600	300
9.6	9,600	9,600	600	300	150
4.8	4,800	4,800	300	150	75
2.4	2,400	2,400	150	75	—
1.2	1,200	1,200	75	—	—
0.6	600	600	—	—	—

TIMERS

Input Frequencies — 1.23 MHz ±0.1% or 3.00 MHz ±0.1% (Jumper Selectable)

OUTPUT FREQUENCIES/TIMING INTERVALS

Function	Single Timer/Counter		Dual Timer/Counter (two timers cascaded)	
	Min	Max	Min	Max
Real-time interrupt	667 ns	53.3 ms	1.33 μs	58.2 min
Programmable one-shot	667 ns	53.3 ms	1.33 μs	58.2 min
Rate generator	18.8 Hz	1.50 MHz	0.000286 Hz	750 kHz
Square-wave rate generator	18.8 Hz	1.50 MHz	0.000286 Hz	750 kHz
Software triggered strobe	667 ns	53.3 ms	1.33 μs	58.2 min
Hardware triggered strobe	667 ns	53.3 ms	1.33 μs	58.2 min
Event counter	—	8.0 MHz	—	—

INTERFACES

MULTIBUS®— All signals TTL compatible

iSBX™ Bus — All signals TTL compatible

iLBX™ Bus — All signals TTL compatible

Serial I/O — Channel A: RS232C/RS422 compatible, configurable as a data set or data terminal; Channel B: RS232C compatible, configured as data set

Timer — All signals TTL compatible

Interrupt Requests — All TTL compatible

CONNECTORS

Interface	Double-Sided Pins (qty.)	Centers (in.)	Mating Connectors
MULTIBUS® System	86	0.156	Viking 3KH43/9AMK12 Wire Wrap
iSBX™ Bus — 8-Bit Data	36	0.1	iSBX™ 960-5
16-Bit Data	44	0.1	iSBX™ 961-5
iLBX™ Bus	60	0.1	Kelam RF30-2803-5 or T&B Ansley A3020 (609-6026 modified)
Parallel I/O	26	0.1	3M 3462-0001 Flat or AMP 88106-1 Flat
Serial I/O	26	0.1	3M 3462-0001 Flat or AMP 88106-1 Flat

MULTIBUS® DRIVERS

Function	Characteristic	Sink Current (mA)
Data	Tri-State	16
Address	Tri-State	16
Commands	Tri-State	32
Bus Control	Open Collector	20

iLBX™ DRIVERS

Function	Characteristic	Sink Current (mA)
Data	Tri-State	9
Address	Tri-State	20
Commands	Tri-State	8
Bus Control	TTL	8

Physical Characteristics

Width — 12.00 in. (30.48 cm)

Height — 6.75 in. (17.15 cm)

Depth — 0.70 in. (1.78 cm)

NOTE: Depth includes a small piggyback on lower left of board.

Weight — 19 oz. (539 gm)

Electrical Characteristics

DC Power Requirements — +5V, 7.0A; +12V, 50 mA; −12V, 50 mA

NOTE: Does not include power for optional EPROM, E^2PROM, or RAM.

Environmental Characteristics

Operating Temperature — 0°C to 55°C

Relative Humidity — to 90% (without condensation)

Reference Manual

145439-001 — iSBC 286/10 Single Board Computer Design Guide (NOT SUPPLIED)

ORDERING INFORMATION

Part Number	Description
SBC 286/10	Single Board Computer

intel

APPLICATION NOTE

AP-114

October 1981

Using the iSBC® 88/40 Measurement and Control Computer in PID Applications

Peter Andersen
OMS Applications Engineering

Order Number: 210263-001

INTRODUCTION

During the past twenty years, the automated process control industry has matured significantly. This is due to the introduction of the digital computer as an element of the control system. At the beginning of this period, the use of the digital computer was limited to a supervisory status in which the actual control was performed by various combinations of relay, analog, and pneumatic systems. Today, systems are off-the-shelf digital hardware and software to perform all the control applications. Indeed, the use of the hardware/software combination has opened entirely new areas of control applications.

The significant increase in computer capabilities and the corresponding reduction in size has been accompanied by a substantial drop in cost. This has led to a strong incentive for users to employ computers in totally new application areas which have resulted from this change in economics. Twenty years ago, few computer control projects were initiated and those which were could only be justified economically in terms of control systems which controlled upwards of 100 loops. Today, a microcomputer system can be justified for a small process which contains as few as 3 or 4 control loops.

Today, the control system engineer's decision is not so much an economic justification of a digital process as it is a choice of whether to use a single or a multiple microcomputer based design.

The trend toward the use of digital technology in the control world has been driven, in part, by the products which have been introduced into the marketplace by Intel Corporation. A recently announced product, the iSBC 88/40 Measurement and Control Computer, is intended to further simplify the implementation of digital technologies into varied control applications and is the subject of this application note. Its architecture is well suited for both single microcomputer and multicomputing environments. The board is also easily adapted to a wide variety of input/output configurations through on-board facilities and iSBX MULTIMODULE expansion boards.

Generalized Computer Application Areas

Those applications in which computers are finding acceptance can generally be broken down into two broad areas. The first involves the acquisition and manipulation of process data by the computer, and is sometimes referred to as being a class of passive applications. The second, known as active systems, also involves the manipulation of the process itself. The systems in the latter class also provide various degrees of passive data manipulation.

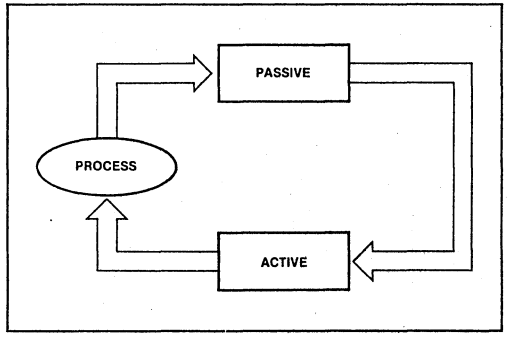


Figure 2. Classes of Computer Applications

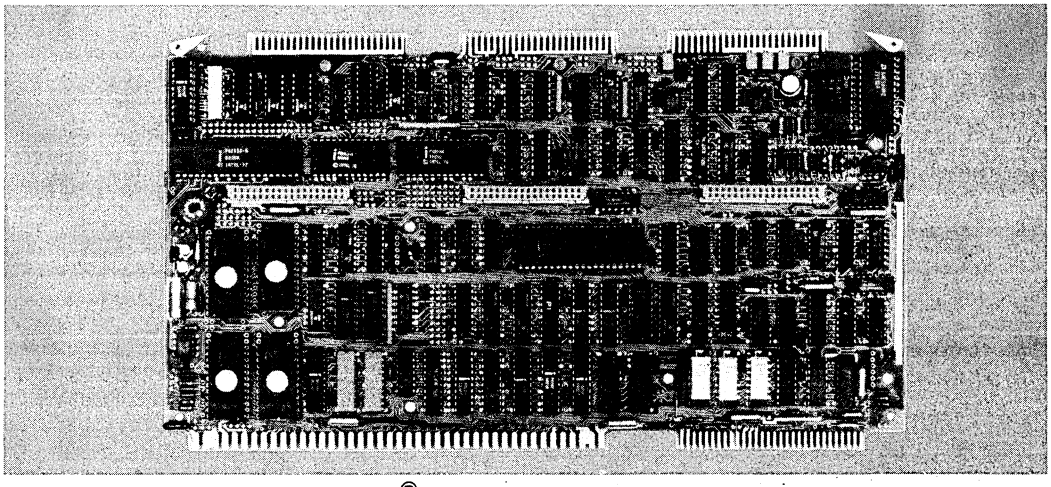


Figure 1. The iSBC® 88/40 Measurement and Control Computer

At first glance, the area of passive computer applications seems to have little or nothing to do with process control; however, many computer design projects are being split into two phases. One phase is to characterize the process and the second is concerned with the actual control system. Many designs never move from phase one and are used as data acquisition systems.

The majority of passive systems involve measuring physical parameters of the process application. Examples are the measurement of pressure, temperature, flow, force, and level. Most transducers associated with these physical parameters provide an analog signal which is proportional to the physical property being sensed. Thus, the ability to measure analog voltages is a requirement of process control systems, both active and passive.

The iSBC 88/40 Measurement and Control Computer is ideally suited for these classes of systems because of the board's built-in analog to digital conversion circuitry. Each input channel (there are 16 differential or 32 single-ended channels on the board) has its own programmable gain which can be software selected to provide full scale inputs ranging from 20 millivolts to 10 volts. The board is thus compatible with most commonly available transducer elements. Examples of typical interface drivers are given in later sections of this application note.

Active applications must interact with the control system in order to manage the process. This normally involves the activation and movement of a mechanical element which is incorporated into the process loop. An amplifier and transducer are required to convert the electrical output of the controller into mechanical energy. The majority of these activators are electro-pneumatic, requiring both an electrical control signal (usually 4-20 milliamps) from the controller and an air supply for its internal pneumatic amplifier. Less common, but still in substantial numbers, are activators which use either a frequency input control signal or stepping motors.

Again, the iSBC 88/40 Measurement and Control Computer provides features designed to allow easy interface to various control actuators. For those actuators using digital frequencies or stepping motors, the board has a parallel output capability to drive up to 24 digital lines. Pulse output signals can be routed from programmable timers/counters (to generate a variety of pulse type outputs) to the external I/O devices. Analog actuators can be driven using the iSBX 328 Analog Output MULTIMODULE Board. This board connects to the measurement and control computer using one of the three iSBX connectors on the iSBC 88/40 board. Each iSBX 328 board can generate up to 8 analog output signals, each of which can function in either a voltage or current (4-20 milliamps) output mode.

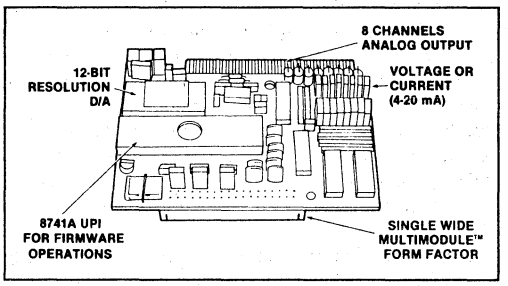


Figure 3. iSBX™ 328 Analog Output MULTIMODULE™ Board

Computer Processing Capabilities

The key to the rapid growth of digital computers in process control has been the flexibility offered by the software. The same hardware can be used in widely varying applications by allowing customization through software programmming. To be successful in the process control marketplace, a digital computer system must be designed in a manner which optimizes the hardware/software relationships. The iSBC 88/40 Measurement and Control Computer does this well.

A powerful instruction set is mandatory if operations are to be efficiently performed by the processor. An instruction set optimized to perform business operations will perform poorly in an industrial process application. The processor used on the iSBC 88/40 board is the Intel iAPX 88/10 microprocessor. This third generation microprocesor is suitable for a wide spectrum of applications. The large application domain is made possible by the processor's dual operating modes and built-in multiprocessing features. The iAPX 88/10 microprocessor is from four to six times more powerful than the 8080A microprocessor.

The high performance of the iAPX 88/10 microprocessor is realized by combining an internal data path with a pipelined architecture that allows instructions to be prefetched during spare bus cycles. Also contributing to performance is a compact instruction format that enables more instructions to be fetched in a given amount of time.

Software for high-performance iAPX 88/10 processors need not be written in assembly language (although it certainly can be). The CPU is designed to provide direct hardware support for programs written in high level languages such as Intel's PL/M-86. Because most high level languages store variables in memory, the instruc-

tion set supports direct operation on memory operands, including operands on the stack. The hardware addressing modes provide efficient implementations of based variables, arrays, arrays of structures and other high level language data constructs. Hardware multiplication and division of signed and unsigned binary numbers, as well as unpacked decimal numbers, is fully supported by the CPU. In all, about 300 forms of machine level instructions are supported by the iAPX 88/10 processor.

Memory Options

A key design requirement for the iSBC 88/40 Measurement and Control Computer was to have the board support a variety of memory types and capacities. The result is a product which can easily be configured to meet a wide range of process control application requirements.

Program storage support for small to very large applications is obtained through the board's ability to include EPROM storage capacities ranging up to 64K bytes. Maximum standard storage capacity is from 8K bytes (using 2716 EPROM devices) to 32K bytes (using the 2764 EPROM). An optional EPROM expansion MULTIMODULE board can be mounted onto the iSBC 88/40 board to double the memory storage capacities.

Variables used in an application are usually stored in RAM memory. A standard on-board RAM capacitiy of 4K bytes is included on the measurement and control computer. In order to efficiently support multicomputer system design, 1K bytes of this memory is dual-ported. Dual-porting introduces a three bus system architecture to system design. An on-board local bus creates a data path between the iAPX 88/10 CPU and its local RAM. Data paths to RAM located on other iSBC boards are provided by the facilities of the MULTIBUS system bus. Finally, a third bus provides a gateway into the local RAM by other MULTIBUS single board computers or bus masters. If additional RAM is required, a small MULTIMODULE RAM expansion board can be attached to the iSBC 88/40 board to add 4K bytes of random access memory.

Even more flexibility can be gained by using unneeded EPROM memory sockets. Because JEDEC standard 24/28 pin sockets have been used, byte wide RAM modules can be inserted into areas of the EPROM memory space. The use of this RAM can considerably enhance the design of certain applications. The board capabilities are such that it is not necessary to have all devices residing in the EPROM sockets be of the same type or size.

Many process control applications require the use of non-volatile memory for the storage of parameter lists and system setpoints. Provision has been made on the iSBC 88/40 Measurement and Control Computer to fully support Intel's new 2816 Electrically Eraseable and Programmable Read Only Memory (E^2PROM). This device gives the user 2K bytes of memory. Depending on the application, up to eight devices (16K bytes) can be used on the iSBC 88/40 board. The board includes all required voltages and wave-shaping circuits to fully support the use of the 2816. A byte of 2816 memory can be programmed in 16 milliseconds. A subsequent section of this application note contains a comprehensive discussion of the operation of the board with the 2816.

Arithmetic Functions

Using computers as an element in a control system leads to extensive arithmetic and mathematical functions. To be effective and attractive to the designer, a computer board must provide a wide range of mathematical capabilities. The iSBC 88/40 Measurement and Control Computer easily meets these needs with varying capabilities for hardware and software functions.

Many applications are adequately handled using the hardware add/subtract and multiply/divide instructions of the on-board iAPX 88/10 processor. Functions needing integer arithmetic of varying precisions are easily programmed using this facility. In some cases, more complex operations may require the use of software libraries to gain the required mathematical functions. The speed and instruction set of the CPU, in conjunction with PL/M-86 statements, make programmers comfortable with these operations.

As processes become more involved and their control algorithms more complex, the need for the processor to support more precise numbers becomes important. The additional precision is usually obtained through the use of a floating point representation. Intel supplies several tools which simplify the implementation of systems requiring floating point operations. Complete support for the floating point numbers is provided as an integral part of the PL/M-86 compiler. Thus, variables can be specified as real numbers. The compiler will perform all numerical operations on these numbers in the floating point format. The data formats of all Intel floating point support conform to the proposed IEEE Floating Point Standard, insuring highly accurate results.

An important feature of the iAPX 88/10 processor is its ability to use a co-processor. The Intel 8087 is mounted on the iSBC 337 MULTIMODULE Numeric Data Processor to provide arithmetic and logical instruction extensions to the 8086 and 8088 CPU's. The instruction set consists of arithmetic, transcendental, logical, trigonometric, and exponential instructions which can all

operate on seven different data types. In many cases, the use of this MULTIMODULE board results in two orders of magnitude performance enhancement over a software solution. This board is the subject of a subsequent section of this application note.

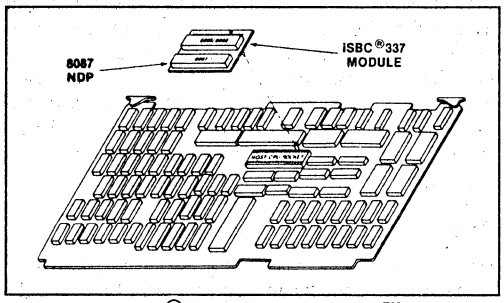


Figure 4. iSBC® 337 MULTIMODULE™ Numeric Data Processor

APPLICATION EXAMPLE

The features of the iSBC 88/40 Measurement and Control Computer can best be shown through an example. This application note describes the classical control system application of an agitated heating tank. Figure 5 shows the prominent features of this process control applications. The process consists of a storage vessel, a temperature sensor which measures the temperature of the fluid leaving the vessel, and a steam coil whose steam flow is regulated by a proportional valve. A motor drives an agitator to insure the temperature of the tank remains homogeneous.

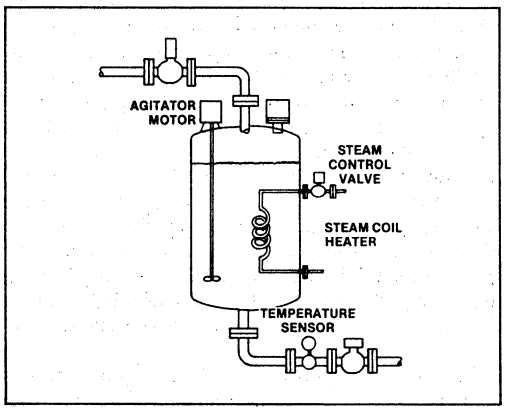


Figure 5. An Agitated Heating Tank

The passive portion of the application involves measuring the actual temperature of the fluid as it leaves the tank (and thus the temperature of all the fluid in the tank). If a control system is to be constructed which will control the temperature, an algorithm must be implemented which will provide control of the steam valve based upon the actual and the desired temperatures. This is the active portion of the application.

The control algorithm selected to control the tank temperature must be capable of compensating for disturbances created by a variety of conditions. For example, the temperature can be affected by changes in steam temperature, input temperature of the fluid, output flow rate, ambient temperature, and the flow rate of steam through the steam coil. Our control system will have control of only one of the variables and will only monitor the output temperature. To gain a degree of stability under these conditions, a feedback control algorithm is required. Alternatively, a system could be implemented using a feed-forward control algorithm. Unfortunately, the latter technique would require extensive instrumentation of all possible variables which could cause a disturbance. A feedback control system can take corrective action regardless of the source of a disturbance. Its chief drawback is that no corrective action is taken until an error is actually detected and, if not "tuned" correctly, some oscillations can occur.

Classical Controller Approaches

Before proceeding with a discussion of how a control system can be implemented using single board computers, a short discussion of classical control system theory is in order. This material will provide a background into the control algorithms which will be used as a basis for the digital control solution which will be developed.

The classical controller for feedback systems uses the "three mode" or PID (Proportional, Integral, Derivative) algorithm. In this system, the control output signal is a function of the error (the difference between the setpoint and the measured system variable). A specific application will use some combination of one, two, or all three terms making up the control statement.

Before continuing with the implementation of the control algorithm on the iSBC 88/40 Measurement and Control Computer, the various terms of the equation will be reviewed.

For Proportional control, the controller output is given by the equation:

$$m(t) = b + k_0 e(t) \qquad \textit{(eq. 1)}$$

where *m(t)* is the output signal, *b* is an adjustable bias value, k_0 is a gain constant, and *e(t)* is the measured error signal. Proportional control systems are normally

not used by themselves since corrections can not be made until an appreciable error has been detected. In addition, they tend to introduce oscillations into the system if the gain is set too large. Another disadvantage of proportional only systems is their inability to maintain a control element at some point (other than at its zero point using the bias term) in the absence of an error signal.

The second term in the PID solution is the Integral. The result of this term is to eliminate steady-state error or offset. The elimination of the offset is an important control objective; thus, the integral control term is widely used in conjunction with the proportional control element. The equation for the integral term is:

$$m(t) = (1/k_1)\, e(t)dt \qquad \textit{(eq. 2)}$$

where k_1 is the integral or reset time.

The Derivative term in the algorithm is used to provide an output which is a function of the rate of change in the error signal. It anticipates the future behavior of the system and improves the dynamic response to the controlled variable by decreasing the process response time. The format for the derivative term is:

$$m(t) = k_2(de/dt) \qquad \textit{(eq. 3)}$$

where k_2 is a constant representing the derivative time expressed in seconds or minutes. Because the output of the term is zero for a constant error, derivative control is never used alone in a control system. Instead, it is always used in conjunction with proportional and integral control. The derivative term is seldom used in flow controllers because derivative control tends to amplify "noise" which is picked up in the flow measurement, leading to an unstable control system. In addition, systems which have very large time delays do not benefit from the use of this term.

Implementation Using Digital Techniques

With an exposure to the fundamental concepts of control theory complete, the development of a solution using the iSBC 88/40 Measurement and Control Computer can proceed. A modular "top-down" approach will be used in this application note. The general requirements will be defined and "black boxes" will be developed to meet these requirements. Finally, the individual pieces will be combined to form a complete solution to the agitated tank control problem.

An effective control algorithm must deal not only with the mathematical solution of the control equation, but must also provide tests on limits and error conditions. As this application note will show, the iSBC 88/40 Measurement and Control Computer is easily able to support these additional requirements.

Additional supporting functions are also needed to effectively implement a complete control system solution. For example, provisions must be made to support input and update of the controller setpoints. Allowances must be made to modify control algorithm constants in order to "fine tune" the system after start-up. Raw analog data must be filtered to eliminate spurious sensor measurements and then must be converted into engineering units. In earlier system implementations not based on digital computers, these functions were performed using a "black box" approach. Here, each function is considered separately and the final solution is composed of combinations of building blocks.

Digital technology offers a simple analogy to this approach. Because application design is performed with software, a "black box" design is available for use with microcomputers. The black box corresponds to a software "task" and the system is integrated into a functional unit using a real time operating system. The iRMX 88 Real Time Executive provides all the tools needed by the software designer to implement his required functions for the application. This application note will show how the iRMX 88 executive can be used to simplify the design and to provide significant features in a process design example.

Figure 6 shows a block diagram of the operations needed to implement the control of one loop for the agitated heating tank. An attribute of using digital microcomputers is that additional loops can be run using the same hardware and software until the I/O or processing capabilities have been exceeded.

Each element of the block diagram represents one function which must be performed by the system. A task will be written to perform the functions assigned to each block. When the tasks are configured together with the iRMX 88 executive, a complete control solution will result. Some key features of the iSBC 88/40 Measurement and Control Computer will now be examined and a typical implementation will be described.

ANALOG SUPPORT FUNCTIONS

The information presented in Figure 6 indicates that many functions involve the manipulation of analog data and its conversion into a digital form usable by the processor. This involves the use of both hardware and software. This section of the application note demonstrates how the iSBC 88/40 board features can be applied to the solution of the analog portions of the system implementation. Both software programming concepts and hardware support products are examined.

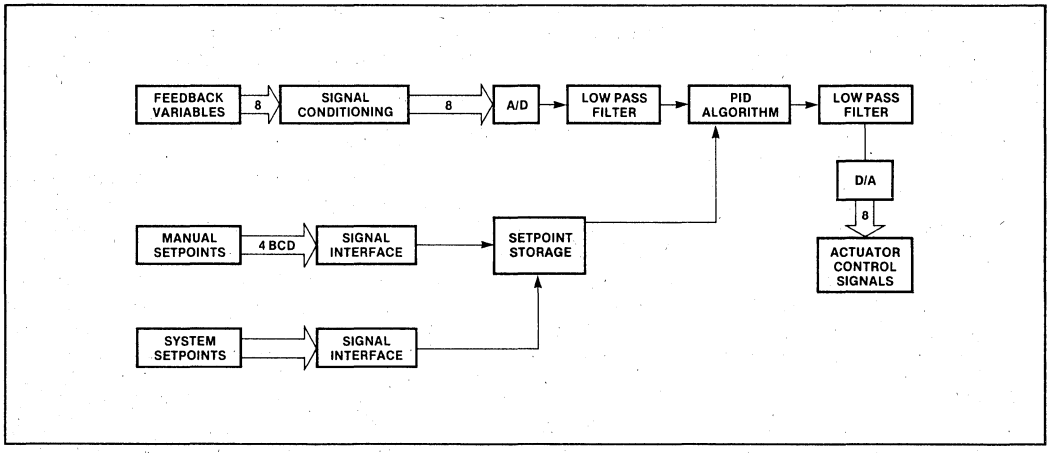


Figure 6. Control System Block Diagram

A digital computer performs most of the control system operations using software. Data is sampled from the process sensor and converted to an equivalent digital format. Subsequent operations use the digital form of the data. Unfortunately, this requirement for operating on sampled data, rather than continuous actual data, can lead to errors if the system is not properly implemented. Care must be taken to minimize errors when the original signal is digitized. Figure 7 shows how the digital signal may look when an analog signal is sampled using an analog to digital converter. A glance at the figure indicates that the error can be minimized by taking samples at shorter time intervals so that the staircase more closely resembles the original signal. Indeed, this is true, but what sample rate is best for a particular input signal?

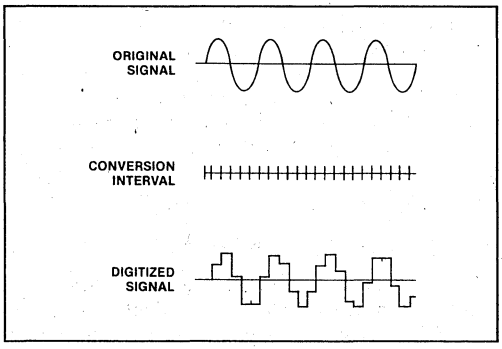


Figure 7. Analog Signal Digitization

A rule for digital control system designers is that the sample must be performed more than twice each period of the original analog signal. Thus, the sampling period must be less than one half the period of the sinusoidal frequency component which must be digitized. Even this method does not, in itself, assure an accurate measurement. Figure 8 shows the effects of the aliasing phenomenon on a high frequency signal. Aliasing converts the high frequency components into fictitious low frequency signals in the sampled results. Before data obtained from a digital system can be used, the unwanted signals must be filtered from the original sensor signal.

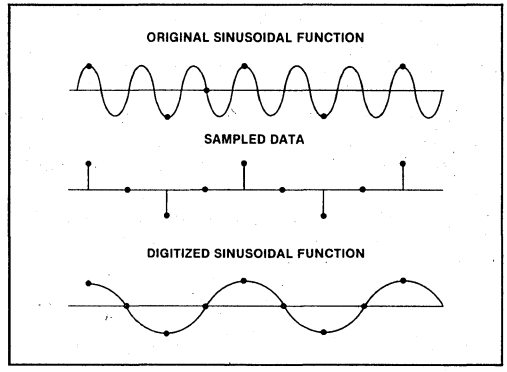


Figure 8. High Frequency Aliasing Error

Two approaches can be considered for filtering the data. One is the creation of an analog low pass filter and the second is the implementation of a digital filter. Unfortunately, a digital filter cannot remove aliasing error and is normally used to provide filtering of very low frequency oscillations. Analog filtering provides effective removal of unwanted frequencies but is expensive when attempting to gain sharp cut-off frequencies. A combination of the two technologies results in an ideal situation when used with digital controllers such as the iSBC 88/40 Measurement and Control Computer.

The final choice of sampling rate is usually determined by examining the process to be controlled. If a mathematical first order transfer function can be obtained for the process, either theoretically or experimentally, then the choice should be to use one tenth of the process time constant. If no function can be obtained and the frequency of the input signal is known and bounded, a sample rate equal to at least twice the input frequency is used. If none of the above is known, a rough estimate for process applications is to use a 1 second sample period for flow measurements, a 5 second interval for level or pressure measurements, and a 20 second interval for temperature or composition measurements. In any case, faster sampling than is necessary is a waste of computing power and limits the number of PID loops that can be supported by a given system.

The elimination of high frequency noise in systems using Intel's control products is best accomplished using the iCS 910 Analog Termination Strip. This strip has provision for the installation of a single pole RC low pass filter (details on the use of this strip in industrial control applications can be found in AP-52, Using Intel's Control Series In Industrial Applications). In addition to providing a front-end low pass filter, the strip gives a simple method of terminating analog wiring to the analog to digital converter. Figure 10 indicates the cable connections which can be used to connect the analog input connectors of the iSBC 88/40 board to the iCS 910 termination strip. This connection arrangement will provide complete compatibility between the numbered channels on the termination strip and those defined by the measurement and control computer.

The iSBC 88/40 board's application software can be used to eliminate the effects of low frequency noise in the sampled signal. This is done by implementing a simple digital low pass filter. The equation for a first order filter is:

$$Sf = a(Sm) + (1 - a)\ (Sf') \qquad \textit{(eq. 4)}$$

where *Sf* represents the filtered output, *a* is a function of the cutoff frequency, *Sm* is the measured sample, and *Sf'* is the last filtered output result. If additional poles are required, the equations can be cascaded as required.

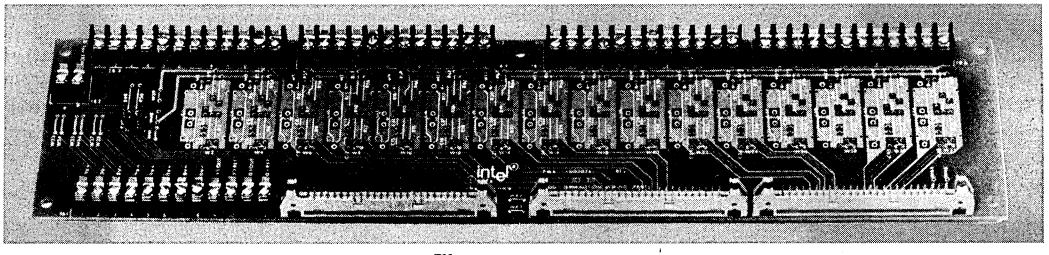


Figure 9. iCS™ 910 Analog Termination Strip

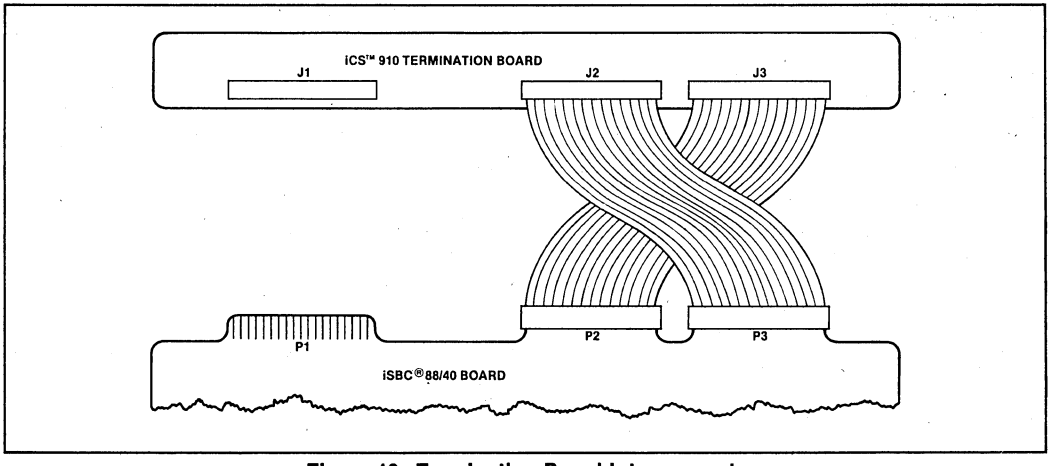


Figure 10. Termination Board Interconnects

The implementation of this filter using Intel's PL/M-86 high level language is straightforward. A simple procedure can be written in which the measured value, the last filter output value, and the value for *a* are passed with the call. The procedure returns the new filtered value. The code for such a procedure is shown in Figure 11. Note that the computation is performed in steps to prevent any stack overflows from occurring when real numbers are used. This should be done whenever the algebraic equation exceeds eight terms. The 8087 stack used in internal operations can overflow when more than eight operations are nested together. Breaking the equation into smaller steps can prevent any overflow errors from occurring.

```
1       Analog$filter$module: Do;
2   1   Analog$filter:
          Procedure (present$value, last$output, cutoff) real public;
3   2     Declare (present$value, last$output, cutoff) pointer;

4   2     Declare New$signal based present$value real;
5   2     Declare Old$filter based last$output real;
6   2     Declare Alpha based cutoff real;

7   2     Declare New$filter real;

8   2     Declare temp1 real;
9   2     Declare temp2 real;
10  2     Declare temp3 real;
11  2     Declare One real data (1.0);

12  2     temp1 = Alpha * New$signal;
13  2     temp2 = One - Alpha;
14  2     temp3 = temp2 * Old$filter;
15  2     New$filter = temp1 + temp3;
16  2     Return New$filter;

17  2   end Analog$filter;
18  2   end Analog$filter$module;
```

Figure 11. Low Pass Filter Algorithm

Before data can be sent to the filter, it must be converted into floating point format and then into engineering units. The conversion into engineering units can involve a complex algorithm if the raw data is non-linear. The design of future systems can be simplified if the programmer generates procedures which are general enough to cover the majority of cases found in his application environment. The following example shows how the iSBC 88/40 board can be programmed to provide the linearization and conversion for the general case.

LINEARIZATION FUNCTIONS

The program developed for this application example uses an interpolation technique. A table look-up enables a program to be written which will support both linear and non-linear analog sensors. The number of entries in the table is a function of the desired resolution and of the non-linearity. For example, linear functions needing only scaling and offset ($y = ax + b$) require only two table entries. A separate table is maintained for each sensor channel. The program is written to support a maximum of 256 entries per channel which should provide at least 0.1 percent accuracy for all but the most non-linear applications.

Each table entry consists of a raw value and a corresponding real engineering unit value expressed in floating point format. The linearization program's declaration of such a table is shown in Figure 12. The application software must determine the bracket or location of the terms in the table which lie above and below the raw input value. The algorithm to find the bracket in the table which corresponds to the raw data input can be programmed as shown in Figure 13. Once the bracket has been found, the actual engineering value can be calculated and passed back to the calling program. The code for performing the interpolation calculation might look like that shown in Figure 14. Data for the tables can be determined from known characteristics of the sensor or a program can be written which allows the user to enter known points into the table dynamically during calibration. In this application note, an assumption is made that the data has been entered into the table from known characteristics rather than actual calibration.

```
6   2   Declare (table based table$pointer)(255) structure (
            x word,
            y real );
```

Figure 12. Declaration of Table

```
10  2   Do while table (n).x < raw$value;
11  3     n=n+1;
          /* special case, above table */
12  3     If n > table$entries
          then do;
14  4       eu=table(n-1).y;
15  4       return eu;
16  4     end;
17  3   end;

          /* special case, below table */
18  2   If n=0
          then do;
20  3     eu=table(n).y;
21  3     return eu;
22  3   end;
```

Figure 13. Bracketing Algorithm

```
                /* interpolate engineering units */
23    2         dx=float(int(table(n).x-table(n-1).x));
24    2         dy=table(n).y-table(n-1).y;
25    2         dr=float(int(raw$value-table(n-1).x));
26    2         eu=dr * dy;
27    2         eu=eu / dx;
28    2         eu=eu+table(n-1).y;
```

Figure 14. Interpolation Algorithm

One final component of the analog design which is required is the creation of software which will actually interface with the analog to digital converter and transform data from the analog world into a digital domain. Again, a program should be developed which is general enough to handle a wide variety of applications. It should be compatible with both the on-board A/D sections and with the iSBC 311 Analog Input MULTIMODULE Board, which may be installed for analog expansion.

The interface with the analog portions of the boards is easily handled using software. The ADC can be commanded to select the desired analog channel and begin a conversion by sending the appropriate byte containing the channel and gain bits to a port corresponding to the ADC. When using the on-board converter, the iSBC 88/40 board user should send the command byte to port 0D8 hex. The actual selection of the desired channel and the conversion takes only 50 microseconds, so little is gained by using an interrupt instead of status testing to detect the end of conversion. The status bit is tested by reading the input status port (0D8 hex for the on-board converter). When the conversion is complete, the bit will have a value of 0.

Certain multiplexer components used in the ADC require that a delay time be added to the basic 50 microseconds for the channel to settle after a new gain setting has been selected before reading the sample and hold converter. The amount of delay is a function of the gain and varies from 0 (gain = 1) to 30 milliseconds (gain = 250). The analog driver software must take this settle time into account. Figure 15 shows the required settle times for the various gain settings. The delay is easily implemented using the facilities of the iRMX 88 nucleus. While the system is waiting for the settling time, other tasks can use the processor to execute their code. Figure 16 provides an example of a program which gets data from the analog to digital converter for a selected channel and gain. The iRMX 88 request for a time delay is implemented using the call to RQWAIT specifying the desired delay. In the example, the system delay increment is assumed to be 5 milliseconds, so the required number of delay increments is specified as 6 in order to wait for 30 milliseconds at high gains. Note that, for gains of one, the delay is skipped. After the required delay has elapsed, the converter is again activated using another output to its command port. This output must again include the channel and gain information.

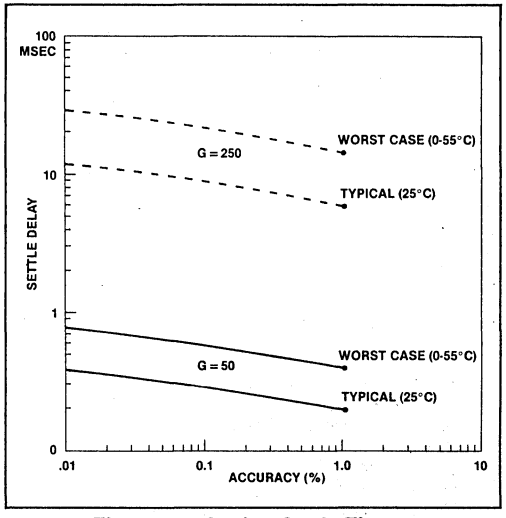


Figure 15. Analog Settle Times

```
                /* select mux channel */
14    2         output(port$adr)=channel or gain;
15    2         if gain < 40h
                then do;
                  /* settling delay for high gains */
17    3           msg$ptr=rqwait(.timeout, 6);
18    3           output(port$adr)=channel or gain;
19    3         end;
                /* wait for end of conversion */
20    2         do while (input(port$adr) and 01h) > 0; end;

                /* get adc data */
22    2         low$raw$data=input(port$adr) and 0f0h;
23    2         high$raw$data=input(port$adr+1);
24    2         raw$data=shl(high$raw$data, 8) or low$raw$data;
```

Figure 16. Analog Input Routine

A workable analog driver must provide more than just the ability to get data from a specified channel. At a minimum, the zero offset induced by the temperature of the circuitry must be removed from the raw data. In some cases, an additional correction is required to compensate for gain error induced by temperature. However, the effect of the latter is small and can usually be ignored.

Provisions are included on the iSBC 88/40 Measurement and Control Computer to simplify the task of providing a zero offset correction. Wire-wrap stakes are mounted

on the board to facilitate grounding one of the input channels. In the differential mode of operation, channel 15 represents the zero reference offset voltage. If a data channel has the offset subtracted from it, the result will be a value which is compensated for offset drift and which is highly accurate over a wide range of board temperatures. Figure 17 shows the software which can be used to collect data from a channel and which will deliver a zero compensated value to a calling program. In Figure 17, note that the values are converted to an offset binary representation to be compatible with the standard output of the analog to digital circuitry.

```
     2   zero$data = get$channel (gain, ref$chan, port$adr);

         /* get data channel */
35   2   raw$data = get$channel (gain, channel, port$adr);

         /* support negative offset */
36   2   if zero$data > raw$data
         then do;
38   3     raw$data = zero$data - raw$data;
39   3     raw$data = 8000h - raw$data;
40   3   end;
41   2   else do;
42   3     raw$data = raw$data - zero$data;
43   3     raw$data = raw$data + 8000h;
44   3   end;
```

Figure 17. Zero Compensation Procedure

The analog input driver required for the application can now be constructed using the software building blocks which have been created. Generally, the input data will consist of either thermocouple inputs or non-temperature sensitive inputs. The driver must be able to support both by providing a selective cold junction compensation correction for those channels which are designated as thermocouples.

The problem is illustrated in Figure 18. The voltage which represents the temperature of the thermocouple consists of the sum of the actual thermocouple voltage plus the voltage which is generated by the thermocouple junctions created where the wiring is terminated. The error introduced by the termination must be removed before a junction temperature can be calculated. If the thermo/voltage characteristics of the termination junction are known, the induced error can be subtracted and the temperature of the thermocouple can be calculated.

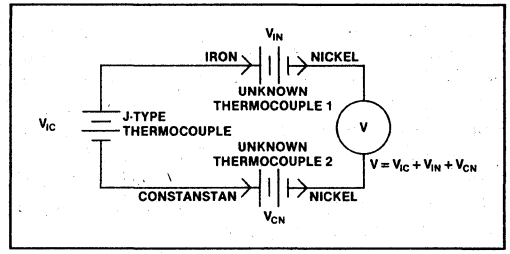


Figure 18. Thermocouple Cold Junction Error

Two things must be known for the correction voltage to become available. First, the actual temperature of the junction board must be known. Second, the electrical characteristics of the junction with respect to temperature must be defined. With this data available, the correction voltage can be obtained using the linearization program which has been created as an analog building block.

The first problem is solved by installing a temperature sensing circuit onto the iCS 910 Analog Termination Strip. Figure 19 shows such a circuit which can be used to provide an extremely accurate measurement of the board and terminator temperature. Note that the circuit is installed onto the termination board using the mounting locations originally designed for the installation of a low pass filter. The output of the temperature sensing is

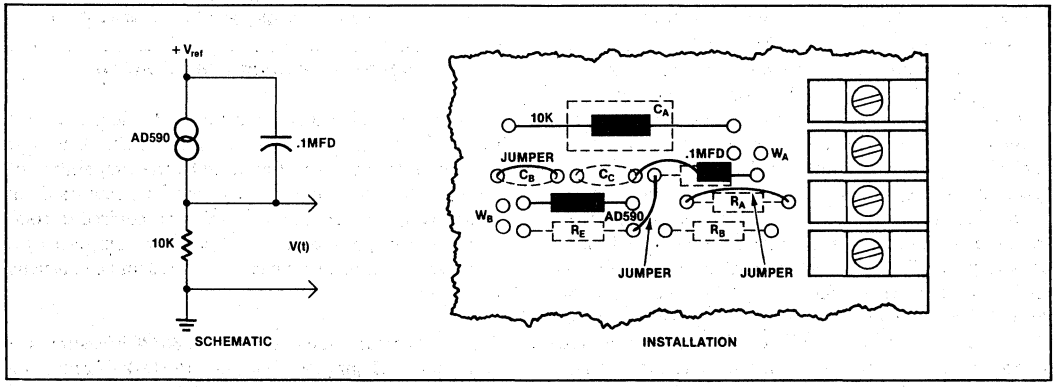


Figure 19. iCS™ 910 Board Sensing Circuit

related only to the temperature of the sensor device which provides a current of 1 microamp per degree Kelvin through the 10K resistor. The temperature is related to the voltage by the equation:

$$V = (273 + T) / 100 \qquad (eq.\ 5)$$

Thus, the voltage read from the termination strip as the temperature varies from 0 to 70 degrees Centigrade will vary from 2.73 volts at 0 degrees to 3.43 volts at 70 degrees. The analog to digital converter should operate at a gain of one to read this voltage. This will provide a resolution (1 bit change) of 0.70 volts / 0.00244141 volts/bit or 286 bits/70 degree change. This equates to about 0.25 degree per bit change.

The second problem is solved by connecting a thermocouple, which is placed in an ice bath, to the iCS 910 strip. The strip is placed into an environmental chamber and the output monitored as the board and junction temperature is varied. The output represents the correction required at each temperature. Tests made for this application note indicated that the error was essentially linear over the board range from 0 to 70 degrees Centigrade. The correction voltage was found to vary linearly from minus 0.102 millivolt at 0 degrees to 3.578 millivolts at 70 degrees. This data was placed into a linearization table to give an offset correction for a measured temperature of the board. Figure 20 shows the table and the code required to correct the raw temperature value from thermocouple inputs.

```
                /* get thermocouple reference junction temp */
60    7         j$raw = analog$to$digital$conversion (
                  gain$one,
                  13,
                  channel$data, port$number );

61    7         tc = analog$linearization (
                  @cold$junction$table,
                  2,
                  j$raw );

62    7         tc = analog$filter (
                  @tc,
                  @channel$data, last$thermocouple,
                  @channel$data, filter$cutoff );
63    7         raw = raw + unsign(fix(tc));
      7         channel$data, last$thermocouple = tc;
```

Figure 20. Thermocouple Correction Program

An analog input driver can now be constructed which is compatible with a variety of applications. It will run as a task under the iRMX 88 nucleus. In order to support up to "n" analog inputs, an exchange is used to store information about current active analog channels. User tasks requiring analog facilities send a request to the analog exchange indicating the parameters of the desired channel. Because an exchange has a FIFO storage capacity for messages, each active channel is sampled by the task in turn, then placed back onto the exchange. A unique message is used to indicate the beginning of the channel requests. Figure 21 provides a partial listing of the code used to make up the analog input task.

```
57    3   msg$ptr = rqwait (.timeout, 2);

58    3   last$channel = false;

59    3   do while last$channel = false;

60    4     msg$ptr = rqwait (.analog$exch, 0);
61    4     if channel$data, type = null$type
            then last$channel = true;
63    4     else do;

              /* test for conversion time request */
64    5       if channel$data, conversion$counter = 0
              then do;

                /* get raw data from adc */
74    6         raw = analog$to$digital$conversion (
                    gain,
                    channel$number,
                    port$number );

                /* perform engineering unit conversion */
83    6         eu = analog$linearization (
                    table$pointer,
                    number$of$entries,
                    raw );

                /* filter the data */
84    6         eu = analog$filter (
                    @eu,
                    @channel$data, last$value,
                    @filter$cutoff );
85    6         channel$data, last$value = eu;

86    6         channel$data, conversion$counter
                = conversion$interval;

87    6         exch$ptr = channel$data, output$exchange$ptr;
88    6         data$ptr = rqwait (exch$ptr, 0);
89    6         data$message, value = eu;
90    6         call rqsend (exch$ptr, data$ptr);

91    6       end;

              /* decriment counter if not ready yet */
92    5       else channel$data, conversion$counter =
              channel$data, conversion$counter - 1;

93    5     end;
94    4     call rqsend(.analgo$exch, msg$ptr);
```

Figure 21. Analog Input Task

Updated data is stored in an output exchange in order to assure mutual exclusion of the engineering unit conversion of the data. Mutual exclusion guarantees that the data cannot be read by another task while it is being up-

dated (during the updating process, multiple bytes of data must be changed; until all are modified, the number cannot be considered valid). The exchange mechanism of iRMX executives supports the movement of messages (this might be compared to a letter in a mailbox). If the data is stored as a message in an exchange, it is available to the first user requesting it. While that user has the message (letter), it is not available to anyone else. When he is finished with it, he will return it to the exchange so that other users may operate upon the data. Note that the sample interval of each channel is selected by the requesting task so that optimum processor efficiency can be obtained.

Certain parameters used by the analog input task must be retained even if the system power is shut off for an extended period of time. These parameters are used to provide the task with unique information such as the channel and port address, the desired gain, the conversion interval and the linearization and engineering conversion data. On the other hand, some information used by the task can be easily created dynamically and does not require the use of non-volatile storage. Examples of the latter category include addresses of the storage exchanges and addresses of the various messages.

The use of E^{2}PROM on the iSBC 88/40 Measurement and Control Computer provides the mechanism for the storage of those parameters which must be occasionally modified. Figure 22 shows a possible technique for passing the analog input task its required information and pointers to the non-volatile data. Intel's PL/M-86 provides a convenient mechanism for referencing variables whose physical location is passed as a parameter. This is the BASED VARIABLE. A declaration is made which indicates the location of the variable containing the address of the data. For example:

```
Declare CONSTANT$POINTER pointer;
Declare CONSTANT based
  CONSTANT$POINTER real;
```

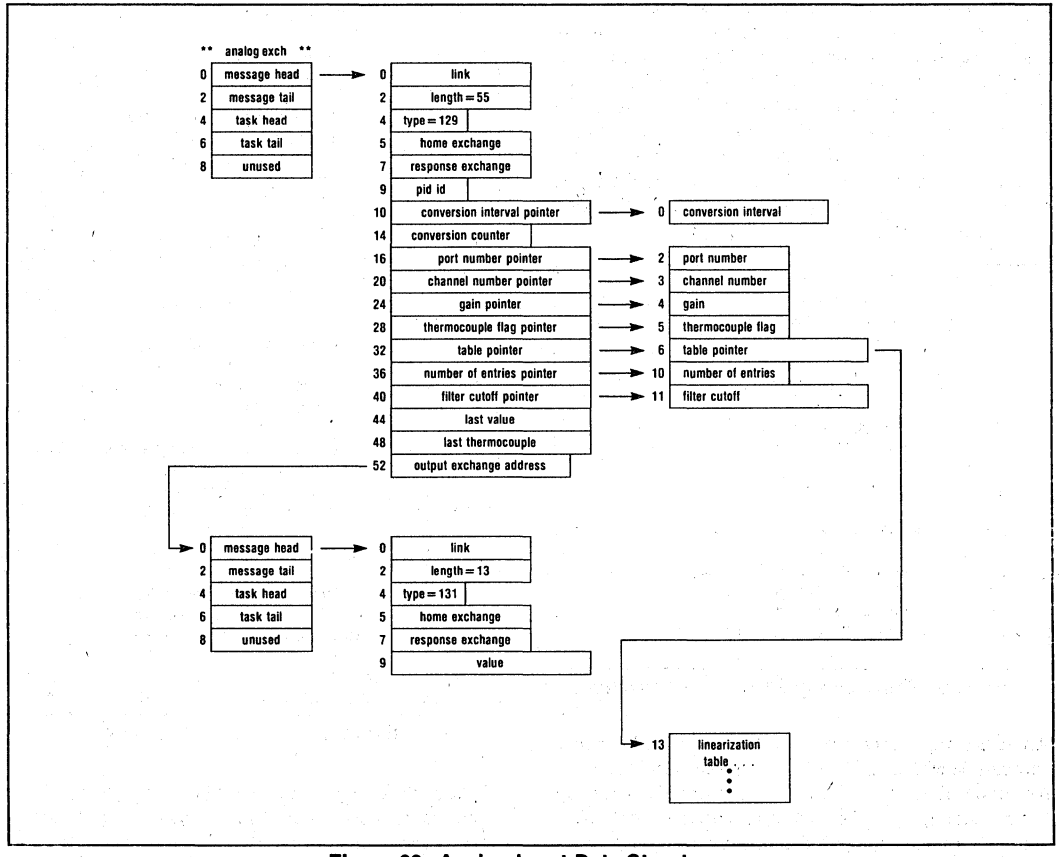


Figure 22. Analog Input Data Structures

The CONSTANT$POINTER contains the address of the constant which is to be used in the calculations. Any program reference to CONSTANT will cause the processor to use the real number stored in the address pointed to by CONSTANT$POINTER.

This technique allows a message to contain pointers to E^2PROM constants which can be used by the task in performing its functions. Indeed, multiple levels of based variables can be used as shown in Figure 22.

CONTROL ALGORITHM

The implementation of a control algorithm on the iSBC 88/40 computer involves more than just implementing the PID equation. To become truly cost effective, multiple loops must be supported by the board and error checking/correction must be included. As with the analog input functions, system control parameters must be maintained in non-volatile memory. Finally, the system must be capable of operating in real time with a minimum of required processor time. This section examines some of the features which are used to provide these functions on the measurement and control computer.

The first design goal is to support multiple control loops using as little of the processor's time as possible. The processing time is minimized by performing all complex mathematical computations using the 8087 math co-processor mounted on the iSBC 337 MULTIMODULE board. Certain details of the software implications of the co-processor are important to the system designer.

In many cases, the iRMX 88 nucleus will provide all the required initialization operations for the co-processor chip. The nucleus sends a default control word setting the device to mask all exceptions and interrupts, define a 64 bit precision, and to round up all operations. In an iRMX environment, the application programmer has no need to send additional mode commands to the processor. However, the mode can be changed by using the PL/M built-in procedure, SET$REAL$MODE, if required. In the application code written for this application note, certain conversion algorithms required that results obtained from the math operations be truncated. To instruct the 8087 to perform this truncation, a command word of 0FBF hex is sent in the initialization segment of a task.

Multiple control loops are implemented using the iRMX 88 exchange mechanism. Here, messages are queued at an exchange in a first in, first out (FIFO) manner. One message can be sent to the exchange for each control loop to be executed. A special message is placed into the exchange at control task initialization to be used as a pointer to the end of the queue. Each time the control task is to run, it will read messages sequentially from the exchange until it encounters its end of queue message. Each message corresponds to one control loop's specifications and is returned to the exchange when the loop has been completed. A separate control interface task manages the control loop activation by sending a message containing the necessary parameters to the control exchange. This technique allows the interface task to also remove a control task by taking the appropriate message from the exchange when parameter modifications or control loop deletion is requested.

Each message at the control exchange (in the application example, this exchange is called PID$EXCH) contains pointers to various other exchanges or data structures. The relationships of these structures is shown in Figure 23. Note that some system parameters should be stored in non-volatile E^2PROM memory. System constants are stored in an exchange pointed to by the primary control message. An exchange is used here so that the system can provide mutual exclusion of the data if it is required to modify one or more of the parameters while the control system is running. Additional exchanges are used to store the input and output terms in order to insure compatibility with the analog input and output tasks.

In order to function correctly, a digital implementation of a PID control algorithm requires operation at a known time interval. In the case of the implementation constructed for this application note, a time increment of 100 milliseconds was desired. The iRMX nucleus provides the ability to perform a timed wait at an exchange via the call to the primitive procedure, RQWAIT. Unfortunately, this procedure can not be directly used in the task to provide the required task delay. This is because the execution time of the task is a function of the number of loops being implemented and also varies slightly depending upon the program paths required by the data values. Thus, a mechanism must be implemented to provide the task synchronization.

The desired time delay can easily be obtained by using an associated synchronization task. In this task, the RQWAIT primitive can be used with the required time delay. Because the task execution time is not a variable, this task can be used to provide synchronization for its supported task. Figure 24 shows how the two tasks can communicate with each other. Two exchanges are maintained. One, called the PID bucket exchange in the implementation, is used by the main task to indicate that it is beginning its execution and that a new time period delay is to begin. The timer task (whose priority should be greater, i.e., having a smaller priority number) will wait at the bucket exchange for the message. When it is received, it will begin a delayed wait at an exchange. When the timeout period has elapsed, the message is sent to a second exchange (in the figure, this exchange is

called the PID trigger exchange). The main task, after completing the servicing of all operational PID control loops, will wait at the trigger exchange for a message from the timer task. In the case of the application example, the message will arrive 100 milliseconds after the task began its last update of the control loops.

When iRMX 88 timed wait operations are implemented on the iSBC 88/40 Measurement and Control Computer, timer 0 of the on-board 8253 programmable interval timer must be used. A wire wrap jumper must be installed to vector the output of the timer to one of the interrupts of the 8259A programmable interrupt controller chip.

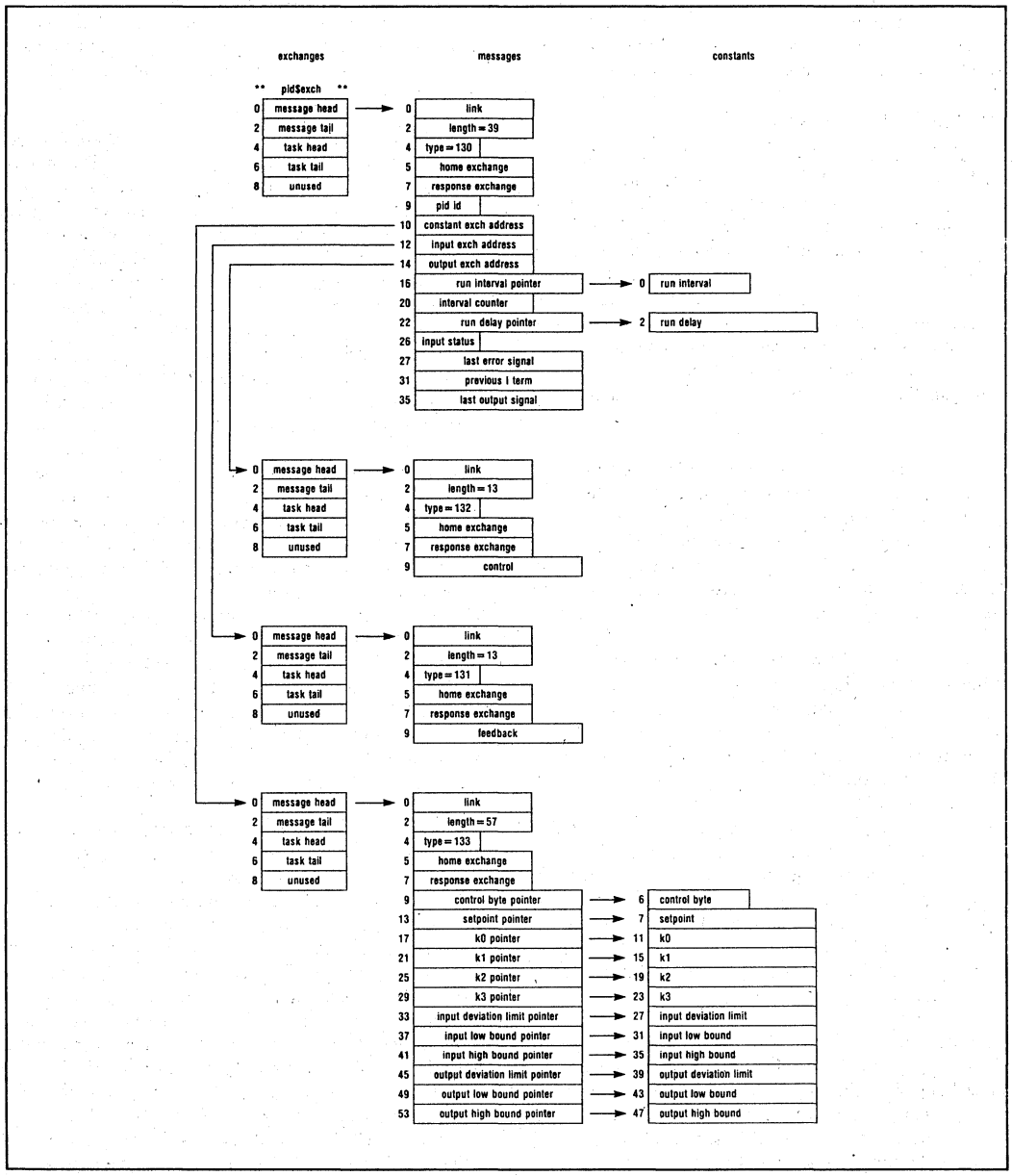


Figure 23. Control Structure Relationships

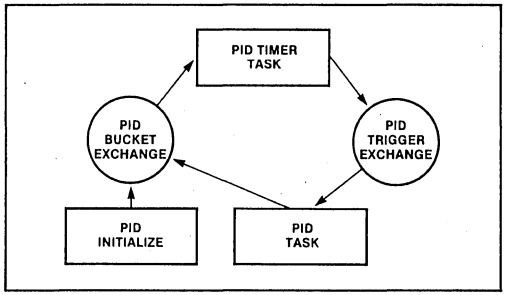


Figure 24. Synchronization Task

Normally, interrupt level 1 is used for this timer; however, any available level may be selected and the iRMX nucleus can be configured to operate correctly by the Interactive Configuration Utility (ICU). A later section of this application note will explain the ICU interaction in more detail. The timed delay function will allow delay increments as small as one millisecond. Each call to the delayed wait function specifies the number of delay increments for which to wait (0 to 65535 increments may be specified in the call). Care should be taken when specifying a delay period of only one unit. The system is not always capable of resolving this delay accurately and an indeterminant delay of from 0 to 1 unit may actually elapse.

ANALOG OUTPUT FUNCTIONS

Once the feedback loop of a control system has been sampled and a control signal generated by the PID control algorithm, a final conversion must be made to provide an output compatible with the system control element. Because such a wide variety of control elements is available, the iSBC 88/40 board was designed to accept the output control circuitry as an expansion option rather than to build unnecessary components and drivers into the board.

In most cases, the control element is driven with an analog signal, either a 4-20 milliamp current signal or a DC voltage. The iSBC 88/40 board is easily interfaced to the analog world using the iSBX 328 Analog Output MULTIMODULE Board. The use of this board is so common with the measurement and control computer that it warrants the time to explain its operation in some detail.

Each iSBX 328 board provides up to eight voltage or current outputs which can drive a wide variety of control devices.

The use of intelligence on the MULTIMODULE expansion board is a key element providing the ability to incorporate eight channels in a very small physical area. The capabilities of the intelligence allow the board to provide significant enhancements to the basic operational characteristics of the host iSBC 88/40 board. One example is the ability to perform diagnostics of the analog output module upon command from the 8088 processor on the host measurement and control board.

The expanded capabilities bring with them a requirement for some care on the part of the system designer. A fixed programming sequence and handshaking are required to reliably communicate with the expansion board. An analog output driver is easily written which provides the necessary support for analog output signals associated with the control application.

Each time a reset is issued to the iSBX 328 board, it executes a test of its internal stored program to assure data integrity and of its usable RAM to insure that each location can be written to and read from correctly. If either of these tests fail, a bit in the status field will be set to indicate the failure. If the test is performed satisfactorily, the F0 status register (bit 2 of the base port + 2) is set to indicate that the board is ready to receive an initialization command. The initialization command is used to specify the operational mode and number of channels being used.

The operational mode specifies which of four internal programs are to be used to move data between the iSBX interface and the outside world. Each program specifies a unique hardware configuration of the board. Two programs are associated with unipolar operation of the DAC outputs. Program 1 is used when some channels are associated with voltage outputs and some are configured as current outputs. Data directed to a current output will be internally scaled and offset before being sent to the DAC. Data specified as directed to a voltage output is not modified by the program. Program 2 indicates that either all eight channels are used for voltage outputs or that all eight channels are used for current output. Current outputs will be offset by the hardware but no scaling is accomplished. This program 2 mode results in a 10% increase in performance over what is specified in the data sheet of the iSBX 328 board.

Both unipolar programs assume that the data is pure binary formatted with a 0 hex corresponding to a voltage level of 0 volts (or 4 milliamps). A value of 0FFF0 hex will generate a voltage of 4.99 volts (in the current configuration mode, program 1 will result in a 20 milliamp current while program 2 will result in an output of 24 milliamps).

There are also two operational modes which can be used to support a bipolar operation. Program 2 provides a direct hardware support capability for those cases where

all outputs are either configured as entirely voltage or entirely current outputs. No adjustments are made to the data prior to being sent to the DAC. The data format used for both bipolar modes is the offset binary representation of a number. Negative numbers are represented by the values 0 (−32752) to 8000 hex (0). Positive numbers range from 8000 hex (0) to 0FFF0 hex (+32752). Channels defined as current outputs have no legal negative output values.

Finally, program 4 is used to support bipolar operations where the outputs are mixed between current and voltage. The program does not alter data destined to voltage channels but does offset and scale data for channels designated as current outputs.

Once the device has been initialized, subsequent data transfers are all through the data transfer port located at the base address of the board's MULTIMODULE socket. Before each write to the device, the driver software must check the IBF bit (bit 1) of the status to verify that the input buffer is not full. An additional bit is used to specify to the host processor which data byte (high or low) is next to be passed. The low order bits of the low data byte specify which channel the data is for and also what configuration (voltage or current) corresponds to that channel.

Like the analog input task, the application driver for the analog output can be an iRMX 88 task using exchanges and messages for data transfer. In the example implemented for this application note, an exchange, DAC$EXCH, was dedicated to the control of the task. It contains messages which specify the output port, channel used, and output mode (current or voltage). The task runs at a twenty millisecond time interval and updates each channel as indicated by the control messages. The location of the exchange used to store the output data is also specified by the control message. The use of this exchange mechanism provides mutual exclusion of the output data.

The external connections to the iSBX 328 analog output board can be made using the Intel iCS 910 Analog Termination Strip. When used in this mode, the analog outputs are available on the terminal strips originally designated as analog input channels. Figure 25 shows how the interconnect cable can be used to install the termination panel to the board.

E²PROM FUNCTIONS

Several references have been made to the advantages gained by using E²PROM 2816 devices on the iSBC 88/40 board for the storage of non-volatile variables. Many configurations mixing E²PROM devices with combinations of EPROM, ROM, and/or byte wide RAM are possible. Support is provided for the installation of one, two, four, or eight (using the iSBC 341 MULTIMODULE EPROM board) 2816 devices. Complete E²PROM write capability is provided on the board. The Intel supplied hardware for this support includes a switching power supply and wave-shaping circuitry. Only minimal user programming overhead is required by the application program.

The on-board wave-shaping circuitry provides a 2816 compatible programming pulse of approximately 16 milliseconds duration. In order to generate this pulse, use is made of the on-board 8253 programmable interval timer. Wirewrap jumper posts are provided to route the timer output to the pulse generator. The gate to the timer is connected using an additional jumper to the memory decode logic to signal a write request to a 2816 device. User software must be provided to program the 8253 for the generation of a 14 millisecond pulse. This code is most easily located in the initialization portion of one of the application tasks associated with writing data into

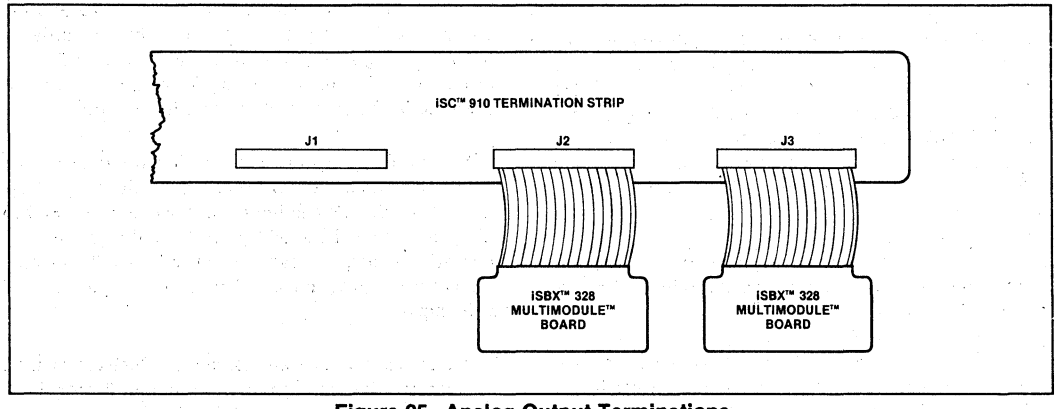


Figure 25. Analog Output Terminations

the devices. Only three lines of PL/M-86 code are required to perform the initialization. The application example includes the code:

```
output(0D6H)=0B2H;
  /* timer 2 to mode 2 */
output(0D4H)=000H;
  /* most significant byte */
output(0D4H)=038H;
  /* least significant byte */
```

The hardware will now generate the appropriate programming pulses to write into the 2816 each time data is written into an address occupied by the device. When the EPROM size is larger than 2K bytes size of the 2816, the system will create a duplicate image of the 2K block as many times as is required to fill the size specified for the EPROM. For example, if 2732A EPROM devices are used and one 2816 is installed at a base location of 0F8000 hex, one image of the E²PROM data will occupy the memory from 0F8000 hex to 0F87FF hex while a second image will be seen from 0F8800 hex to 0F8FFF hex. Reads or writes to either image will access the same data and either may be used.

The user must consider the possibility of system power failures and their impact when designing systems which use the iSBC 88/40 board's E²PROM capabilities. This is especially true in systems whose power supply for the +5 volt source is protected by a crowbar circuit. The onboard switching power supply which generates the high voltage programming pulse operates at very low input voltages and its RC time constant will provide significant voltage levels even if the +5 volt input supply is abruptly removed. The presence of a programming voltage in the absence of a +5 volt supply to a 2816 can cause irreversible damage to the E²PROM chip. The potential for this condition during a write cycle must be considered by the designer. Figure 26 shows a circuit which can be added to the system and connected via the iSBC 88/40 board's P2 connector if desired. The purpose of the circuit is to crowbar the V_{pp} programming voltage to the +5 volt supply if the +5 volt voltage level drops below about 4.5 volts, thus preventing any damage to the 2816.

From a software standpoint, only two items need be given attention during the writing of E²PROM devices on the board. First, before any location can be written in the 2816, the location must first be cleared to an initial value of 0FF hex. Unless this value is already present in the device, two write cycles are required to store new data (the 2816 has a chip erase mode but it is not supported on the iSBC 88/40 board). The second item involves inhibiting interrupts during the write cycle. The programming pulse generation circuitry uses the onboard timeout circuitry, so the timeout interrupt, if used, must be masked off prior to beginning the write cycle (this implies that the hardware for the timeout acknowledge must be installed to all the circuitry to become a part of the pulse generator). In an iRMX 88 environment using timed waits, the interval timer must also be masked off during the write cycle. If these interrupts are not masked off, the processor can respond to an interrupt and begin modifying its internal registers which point to the memory. This will result in incorrect programming of the E²PROM device. An example of the code which might be used to program a 2816 is shown in Figure 27. The programmer should keep in mind that, during the programming of the 2816, the iAPX 88/10 processor is in a wait state and cannot process any instructions. Thus, for each byte written, approximately 36 milliseconds must elapse before processing can again begin (18 milliseconds for the clearing of the byte and another 18 for the data write). If timed waits are being performed, an error will be introduced into the system. Some critical applications may need to take this into account.

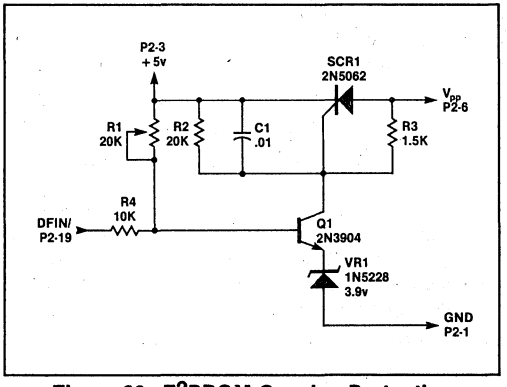


Figure 26. E²PROM Crowbar Protection

```
                then do;
366   5           disable;
367   5           call movb(@erase$pattern(0),
                  @constants(k).table$pointer,4);
368   5           constants(k).table$pointer=
                  @constants(k).linearization$table(0);
369   5           enable;
370   5         end;
```

Figure 27. E²PROM Programming Example

System Implementation

The application programs for the example described in this application note have been implemented using Ver-

sion 1.1 of the iRMX 88 executive. The use of the Interactive Configuration Utility (ICU88) considerably reduces the effort required to bring a system on line by providing a question and answer session with the programmer. The output of the ICU consists of a system configuration module and a submit file which provides most of the required LINK and LOCATE commands.

In the application, a system time wait increment of 5 milliseconds was chosen. Figure 28 shows the dialogue required to implement the timed wait feature using the board with the output of channel 0 (from the 8253 programmable interval timer) connected to interrupt level 2. Note that the system time unit of 6140 corresponds to a 5 millisecond increment.

```
INTERRUPTS: Y
  8259A PORT: C0H
  8259 INTERVAL: 2
  INTERRUPT SERVICE ROUTINE VECTOR BASE: 56
  TIMED WAITS: YES
    TIMER LEVEL: 2
    8253 PORT: D0H
    SYSTEM TIME UNIT: 6140
```

Figure 28. ICU Timed Wait Dialogue

Additional entries into the ICU define the system tasks and their associated exchanges. The desired locations of the RAM and EPROM are specified and the configuration modules created. The location of the E2PROM module is specified in one of the user created modules as a public pointer which is initialized with the base address of the device. An example might be:

```
e2prom$module: do;
  declare e2prom$pointer pointer public
    data (0f8000h);
end e2prom$module;
```

All references to the data structures in the 2816 are by means of a based variable or structure.

Before executing the submit file for a ROM based system, it is necessary to edit the LOCATE command to include the BOOTSTRAP request. This will assure that the locator places a long jump at the reset vector location when the system is executed out of EPROM. Execution of the LOCATE facility will generate a warning 38 which should be ignored. The corrected LOCATE code is shown in Figure 29.

```
ISIS-II MCS-86 LOCATER, V1.3 INVOKED BY:
LOC86 :F1:ADCINP.LNK TO :F1:ADCINP   MAP
 PRINT(:F1:ADCINP.MP2)&
BOOTSTRAP ORDER(CLASSES(DATA,STACK,CODE))&
ADDRESSES(CLASSES(CODE(0FC000H), DATA(000400H)))
WARNING 38: SEGMENT WITH MEMORY ATTRIBUTE NOT PLACED
 HIGHEST IN MEMORY SEGMENT: MEMORY
```

Figure 29. LOCATE Modification

The total control system for the application example can now be assembled using the hardware and software discussed in this note. The same "black box" approach used in hardware designs can be extended to include both software and hardware implementations. Figure 30 shows the complete solution to the control of up to eight agitated heating tanks.

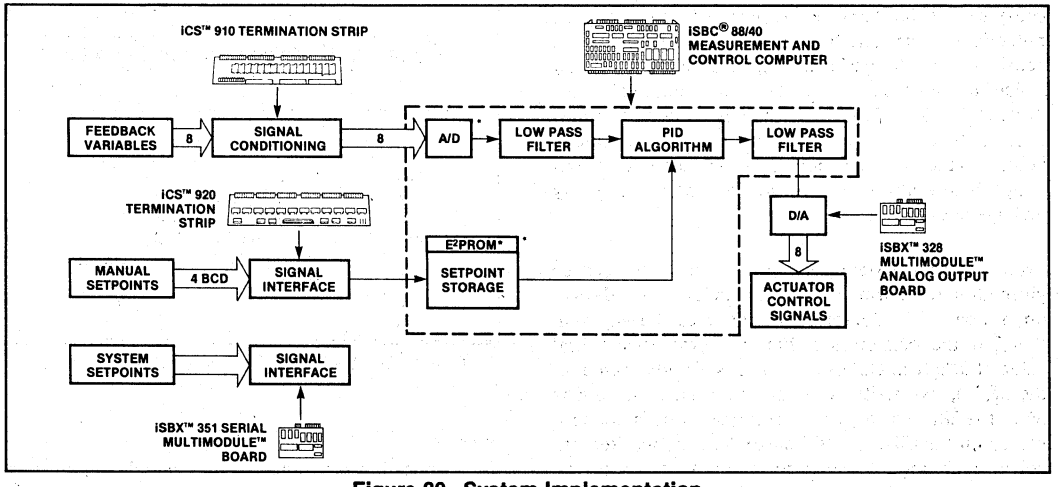


Figure 30. System Implementation

CONCLUSIONS

The purpose of this application note is to illustrate how the Intel iSBC 88/40 Measurement and Control Computer can be used to solve a complex control application. This has been done in Intel's lab and the results obtained from operating the board indicate that the system performance is sufficient to support the operation of eight loops each 100 milliseconds. Observed operation of the analog input and engineering unit conversion task indicated a 4 millisecond per channel execution time. The actual PID code required only 5 milliseconds per loop to execute.

The ease of implementation and fast execution time for multiple complex loops is a result of many Intel product features. For example, the use of the iRMX 88 Real Time Executive provided fast, small, and easy-to-use multitasking real time software for use on the single board computer. The iSBC 337 MULTIMODULE Numeric Data Processor Board enabled the use of high accuracy, easy-to-use floating point calculations without taking excessive execution time. If desired, a fixed point integer math algorithm could have been substituted for the floating point without changing the system performance appreciably. The iSBX 328 Analog Output MULTIMODULE Board provided low cost customization of the base board to support a variety of controllers and actuators.

Finally, the use of the Intel 2816 E^2PROM provided the non-volatile storage for system setpoints and constants which is required in a control situation.

Above all, the iSBC 88/40 Measurement and Control Computer provided a platform and execution vehicle for mounting and operation of the various ancillary features. Its iAPX 88/10 processor, memory and MULTIMODULE expansion sockets provided the flexibility to easily customize the board to a particular application environment.

APPENDIX A

APPLICATION CODE AVAILABILITY

The programs which were used to construct the application example are available from Intel through Insite. Insite, Intel's Software Index and Technology Exchange, is a collection of programs, subroutines, procedures and macros written by users of Intel's 8008, 8080, 8085, 8086, 8088, and 8048 microcomputers. Information on how to join Insite and obtain the source code can be obtained from your local Intel sales office or distributor, or by writing to:

North America

Intel Corporation
User's Library 6-5000
Microcomputer Systems
3065 Bowers Avenue
Santa Clara, California 95051

Europe

Intel International Corp. S.A.
User's Library
Rue du Moulin a Papier 51
Boite 1
B-1160 Brussels, Belgium

Orient

Intel Japan K.K.
User's Library
Flowerhill-Shinmachi, East Bldg.
1-23-9 Shinmachi, Setagaya-ku
Tokyo 154, Japan

intel

ARTICLE REPRINT

AR-229

July 1982

Enhanced μC Boards Strengthen Factory and Office Controllers

Gary Sawyer
Scott Tetrick
Don Peterson
Electronic Design

Order Number: 210825-001

New single-board computers sport dramatic advances in density and speed. The upshot: much more capable factory and office controllers.

Enhanced μC boards strengthen factory and office controllers

Designers of control and computer systems for the factory or office should be aware of the updated performance capabilities of the latest breed of single-board microcomputers. Faster microprocessors with wider word lengths are joining with denser, faster static memories to effect even finer control of time-critical operations. Yet the new crop maintains bus and software compatibility with earlier designs, needs fewer power supplies, and remarkably, charges no penalty for the extra work. In some cases, costs will dip.

Using such boards as the iSBC 86/XX family, designers can trade in 8 bits for 16 (see "A Bridge from 8- to 16-bit Processing"), 5-MHz operation for 8, older memory sockets for standard 28-pin JEDEC EPROM sockets. In addition, plug-on modules offer low-cost expansion. That adds up to double the EPROM capacity with no redesign and to quadruple the RAM capacity.

Moreover, the iSBX 86/XX boards support the full IEEE-796 Multibus specification—including 16-Mbyte addressing features (send and receive) and the use of the lock feature (Fig. 1). Four additional address lines on the Multibus can be used to address memories larger than 1 Mbyte. The lock function increases system speed—by preventing arbitration delays—and serves in semaphore signaling applications (for more particulars, see "Meet the Family").

In a multiprocessing system, individual processors must have a method of synchronization and mutual exclusion. A semaphore is the most common software method for providing these functions. Figure 2 shows an example of the semaphore concept and the need for dual locking. Assuming as in the figure that the system consists of two processors and a disk-controller board, only one processor can access the mass-storage controller at any given time. A global memory byte indicates whether the disk-controller board is in use—a 1 indicates not busy; a 0, busy.

When a request is made for a disk file, the requesting processor must check the global byte to determine whether the controller is busy. Although the process appears simple, there could be pitfalls (Fig. 3a). The hardware must allow the processors to read the semaphore bit without the intervening write cycle, to prevent deadlocking.

Earlier boards needed global memory, not dual porting, to support semaphores easily. They relied on Multibus arbitration to prevent deadlock, but that required an additional memory. Moreover, the boards required much more hardware and software to ensure proper operation. With the introducton of the IEEE-796 Multibus specification, a lock line added to the system bus allows a much simpler implementation of semaphores in a dual-port memory (Fig. 3b).

The on-board processor can prevent access to the dual port, and system accesses can lock the local processor out of the dual port. Both functions must be implemented to ensure the integrity of the semaphore byte. Figure 4 shows the arbitration scheme for a dual port.

Other board enhancements include Schmitt-trigger inputs on all critical Multibus command lines, which improves noise immunity. Pull-up resistors on all input lines allow for easy testing. The new board layouts minimize the length of the Multibus signal lines, reducing the line's susceptibility to crosstalk pickup. These improvements result in a mean time between failures (MTBF) that is 50% greater than that of comparable first-generation boards.

Most measures of a computer board's processing power are summed up in one specification, the clock

Gary Sawyer, Product Manager
Scott Tetrick, Product Engineer
Don Peterson, Marketing Engineer
Intel Corp.
5200 N.E. Elam Young Pkwy.
Hillsboro, Ore. 97123

rate. With an 8-MHz 8086-2 microprocessor on board, processing speeds up 60% over 5-MHz operation.

Because of increased processing speed, on-board circuitry must be upgraded to handle higher data and instruction rates. Thus, some control functions are implemented with programmable logic arrays (PLAs). These high-speed devices not only replace dedicated chips, but also allow logic operations to be performed at clock rates exceeding 10 MHz.

Stepping up the speed

To accommodate the demand for greater data handling, RAM access time is 750 ns, compared with 1000 ns in earlier single-board computers. An additional benefit of faster access time is the capacity for dual-port access. With this feature, dual-port RAMs can be accessed from the Multibus in 500 ns when locked and 800 ns when unlocked. High-speed, dual-port access permits interprocessor communications to occur at a much faster rate.

Since each iSBC board contains a different RAM storage capacity, speed performance is linked closely to a board's memory capacity. As Fig. 5 shows, numbers on the vertical scale indicate increasing performance for a given storage capacity.

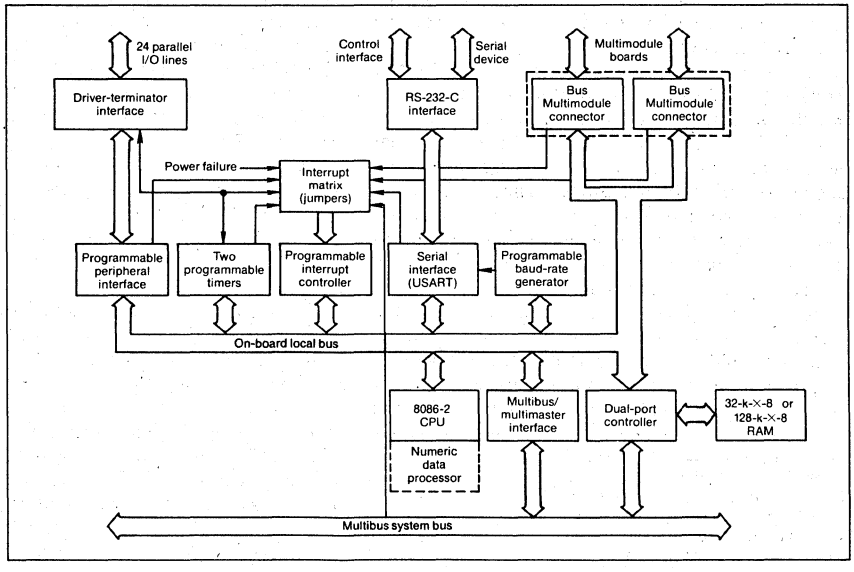
The timing/storage performance graphs of Fig. 5 have been determined by making certain assumptions about system operation. First, it is assumed that the system has no Multibus contention for the global memory as is the case in a single-master system. Also, all memory is accessed constantly, independent of on- or off-board optimization. For example, because of delays encountered in acquiring the Multibus, data rates plunge (in the iSBC 86/05) if more processors are added to a system.

The expanded memory space—320 kbytes—can serve either for enhanced multiprocessing or for the storage of large amounts of application code. The new design prevents performance-reducing arbitration delays in multiple-processor system operation. Proper partitioning of the system memory allows code accesses from the on-board memory. Since code fetching operations occupy most of the system-bus bandwidth, execution from local memory greatly accelerates system throughput.

At home in factory or office

Two key application areas—and their diverse requirements—stand out for single-board computers. One area includes machine and process control, instrumentation, and specialized data acquisition. Here, high-performance single-board computers are required to have extensive EPROM capacity to provide sufficient program storage without relying on mass storage.

The other area covers communications systems, small-business computers, and word and data processing. Single-board computers in office-of-the-



1. Advances in digital LSI and VLSI have expanded the capabilities of single-board computers. The result is more speed and more memory, yet compatibility with earlier versions.

future applications are generally RAM-intensive, with programs downloaded from a disk. A small amount of EPROM serves to bootstrap the system RAM at power-up. Mass storage is used to hold the large data bases required, but the computer board must support an I/O terminal for the human interface. Typically, word-processing systems require a high-performance CPU, a large RAM capacity, an interface for mass storage, a CRT terminal, and a printer.

For RAM-intensive applications, a 128-kbyte RAM module fits on board. An on-board serial I/O port and 24 programmable I/O lines provide the interface with the CRT display and printer. With the inclusion of the iSBX bus on the board, low-cost I/O expansion is possible. Also, the iSBX 218 single- or double-density, single- or double-sided floppy-disk controller allows the entire system to fit on a single board. Moreover, even with those components on the board, there is room for a mathematics module. So the same basic system can serve as a high-performance small-business computer.

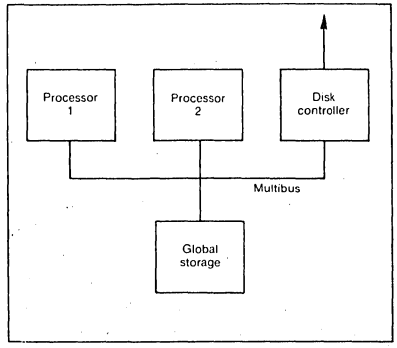
In high-speed communications applications, a high-end single-board controller outfitted with an iSBX module offers an interface capable of handling processing needs well into the future. Since the iSBX module communicates directly with the CPU over the iSBX bus, Multibus contention does not pose a problem. Another benefit of the bus-module approach: reduced size and power consumption.

Factory-control considerations

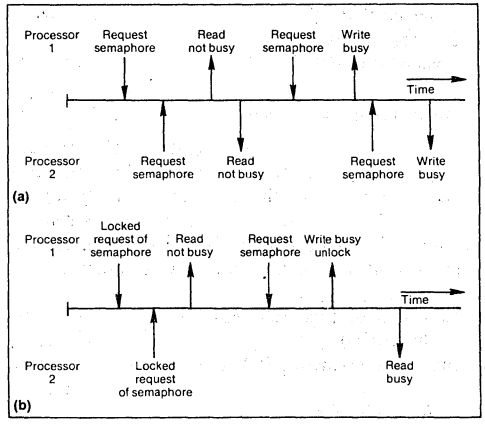
When the single-board controller is to be used to control various phases of a manufacturing process, two things must be considered. First, operations are time-critical. The time between samples in a manufacturing process defines the maximum interval during which defective products could be made. Often, the volume of production requires the use of multiple processors.

Second, factory operations do not occur at predetermined intervals, but are event-driven. In addition, code operations are usually fixed, since the factory environment is quite predictable. Thus, code can be stored in PROM or EPROM, but that requires enough ROM storage capability on the controller board. Finally, specialized operations are common, the most often used being floating-point arithmetic for three-dimensional movement and analog input and output for manipulating machinery.

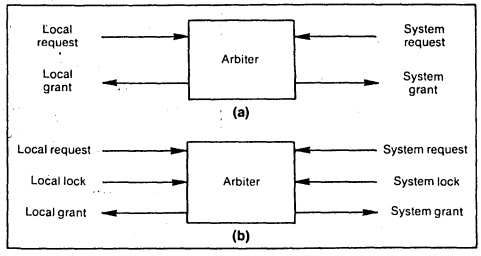
To equip a general-purpose board as a factory controller, its interrupt structure, PROM, and I/O access must all be high-speed, and the board must have a range of configurable options. After selecting a board, the designer should determine his maximum interrupt-speed requirement by measuring two



2. Semaphore signaling is a method of synchronizing the operation of processors in a multiprocessing system. In an earlier single-board system (shown here), global storage provides the control signal for indicating the status of the disk controller—busy or not busy.



3. In newer single-board computers, a locking feature eliminates the need for global storage in semaphore signaling. Without locking, a deadlock can occur (a) when two processors try to access a system resource. The dual-port memory prevents such a problem (b).



4. Adding dual-port memory locking to computer boards inhibits arbitration until the locked operation is completed (a). Earlier boards relied on global storage (b).

things: the interrupt delay time, or the maximum time allowed from an interrupt request until the interrupt service routine instructions are executed, and the maximum number of interrupts that may need servicing in a given period.

To compute interrupt delay, both hardware and software must be considered. For example, hardware on the board controls the processor's acknowledgment of the interrupt request. The board's 8086 processor retrieves the address of the interrupt-service routine from the interrupt vector table, which is stored in an on-board RAM. Interrupt-service routines often store all registers on the stack before the servicing of an interrupting device begins. The total delay time is computed as follows:

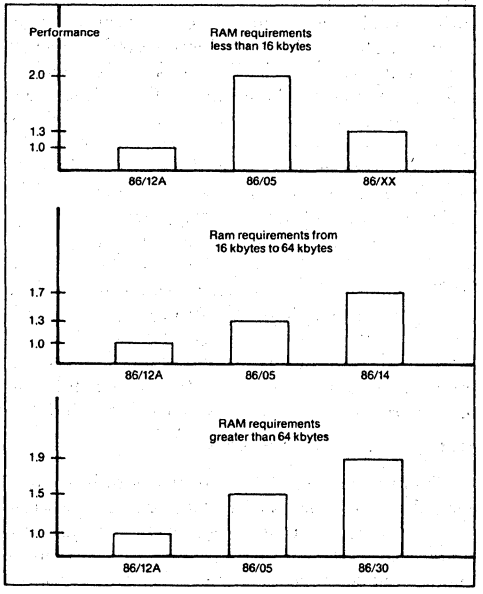
INT to 8259A to INTR at 8086	350 ns
INTA service time (16 8086 clocks)	2000 ns (hardware)
8086 INT vector computation (56 clocks)	7000 ns (processor)
Storage of nine 8086 registers (103 clocks)	12,875 ns (software)
Total interrupt delay time	22,225 ns

The interrupt frequency depends on the interrupt delay time, the execution time of the interrupt-service routine, and the time to restore program execution. Time spent executing the service routine is highly device-dependent, but the return to program execution can be computed as follows:

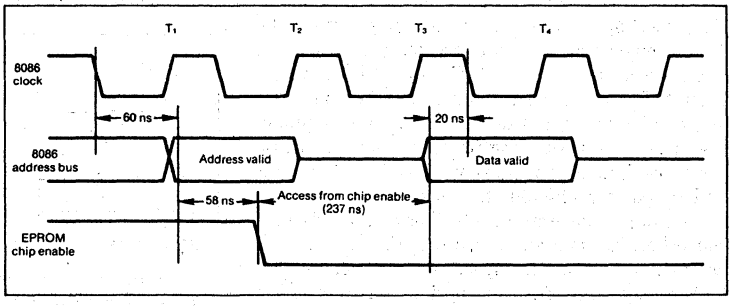
Retrieval of nine 8086 registers (90 clocks)	11,250 ns (software)
Interrupt return of 8086 (30 clocks)	3750 ns (processor)
Code fetch of next instruction (four clocks)	500 ns (hardware)
Total return time	15,500 ns

Interupt frequencies for the iSBC 86/14—assuming several I/O service routines—are 24,600 per second for sending a character to a CRT, 24,600/s for receiving a character from the CRT, and 22,000/s for clock interrupts.

The access times of current EPROMs range from 200 to 1000 ns. For slower devices, wait states (an integer number of processor clock times) are added to the access time. At an 8-MHz processing frequency, a 200-ns EPROM operates with zero wait states. The most critical time, access time from a chip



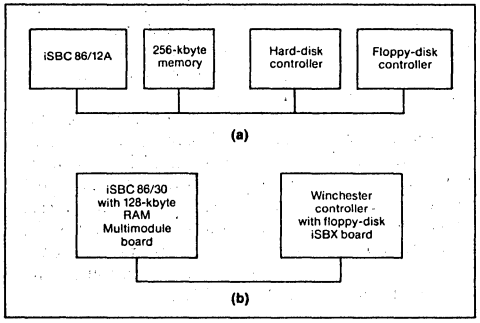
5. The random-access-memory performance of single-board computers is related to the amount of storage a board provides. The latest boards are designed to work most efficiently with storage capacities of 16 to 256 kbytes.



6. Factory controllers execute most of the system code from EPROMs. With a high-speed EPROM, valid data appears on the Multibus about 300 ns after valid addresses are applied.

enable, is 200 ns. The timing relationships for accessing an EPROM are shown in Fig. 6.

From a system standpoint, all peripheral components can be accessed in about 300 ns after a command. At a clock rate of 5 MHz, that forces one wait state; at 8 MHz, two wait states. A variety of devices interfaces with the iSBX bus, permitting custom configurations of a single-board controller for factory applications. Two analog modules, one for input and one for output, can monitor or control plant equipment. For expansion of digital inputs or outputs, a parallel I/O module can be added.



7. Considerable hardware can be saved when a single-board computer replaces earlier versions (a) in an office controller. One advantage of the new boards (b) is their large on-board storage capacity, which allows code to be stored locally, increasing system speed and throughput.

Whereas interrupt response time is a critical parameter in a factory environment, other considerations become critical in an office environment. One of those is throughput, the number of tasks a worker can perform. As a result, office controllers must provide the user with fast response time. In addition, since office controllers usually perform a wide variety of tasks, execution is mainly from RAM, not ROM. Moreover, because most tasks are dynamic, large mass-storage devices are needed. Programs are loaded from mass storage into the processor's memory and executed. In addition, most programs are in a high-level language (for word processing and other office tasks), which requires considerable hardware and software support.

As with factory controllers, configurable systems are important in the office environment, since special applications will require additional hardware. Word processors, for example, are generally linked via a serial communications line to a central processor, or mass-storage device, and the link must be a high-speed one to minimize response time.

To meet office system requirements, computer boards should have the following features: an operating system having a human interface (keyboard), a program loader and configurable peripheral support for mass storage, communications controllers and high-level languages; a maximum-size RAM with minimum access-time devices for program support;

A bridge from 8- to 16-bit processing

Until recently, most dedicated controller applications had to be handled with an 8-bit microcomputer because of price and performance considerations. But with performance and software needs on the rise, designers are faced with moving up to 16-bit devices to obtain the necessary computing power. When the move up happens, however, hardware and software compatibility considerations often get the short end of the stick.

The iSBC 88/25 could be termed the missing link between 8-bit buses and 16-bit processing power. Using the 8088 microprocessor, the board interfaces with the outside world on an 8-bit bus, but internally it operates as a 16-bit system. Since software compilers such as the PL/M-86 often generate 16-bit data-access instructions, a designer using a 16-bit 8086-based board must have a 16-bit memory expansion board or pay a burdensome software penalty. What's more, an 8086-based board must access instructions in 16-bit words, eliminating the possibility of using 8-bit boards for instruction storage.

The iSBC 88/25 automatically breaks all 16-bit words into 8-bit bytes on the Multibus. Therefore, independently of software, the hardware converts all accesses to 8-bit bytes transparently to the user. This gives the board the dual advantage of being fully compatible with all 8-bit hardware devices and 16-bit software. To retain compatibility with Intel standards, the board is designed to interface with the Multibus system bus, the iSBX bus, memory modules, and the iSBC 337 math module and numeric data processor. The board also contains JEDEC-compatible 28-pin sockets to accommodate EPROM and static byte-wide RAM.

The 8088 microprocessor is being used in the second generation of personal computers, such as the one from IBM. Several other personal-computer and small-business systems will also use the 8088. With its software compatibility with the 8086 processor and widespread application in personal and small-business systems, the software repertoire of the 8088 will exceed that of any microprocessor.

and hardware support for both mass storage and communications.

Office controllers call for lots of RAM, and a board with a capacity of up to 256 kbytes is suitable. A dynamic RAM controller handles the automatic refreshing and generates the proper timing for the row and column addresses. PLA devices send commands to the controller to initiate the RAM access.

The differences between early and updated office controllers are shown in Fig. 7. Additional memory expansion on the Multibus is necessary to meet system requirements. The floppy-disk controller board alone must be implemented with a full size Multibus board. Using newer boards not only reduces system size, but also improves performance. The large on-board memory stores enough program information to allow operation that is 100% faster than first-generation boards.□

Meet the family

Progress in semiconductor density and speed translates into more functions and greater throughput in the latest breed of single-board computers. Another, more subtle, benefit accrues: board-level controllers can be made to fit within a wide range of performance and cost slots, resulting in greater freedom of design choice.

Thus whereas one board, the iSBC 86/05, provides high-speed, low-cost compatibility with earlier units, two other boards, the iSBC 86/14 and iSBC 86/30, drop into the high-performance end of system integration. Both controller boards are memory-intensive: the former is available with 32 kbytes of RAM, expandable to 64 kbytes; the latter starts with 128 kbytes of RAM and can be expanded to 256 kbytes. Large storage capacity is backed up by 8-MHz processing rates and iSBX-bus support.

Since all 16-bit single-board controllers are software-compatible, both present and future controller systems can be upgraded with minimal extra investment. Once established and proven in prior designs, bus and software standards reduce hardware conversion costs.

Since iSBC 86/XX boards constitute a family, hardware and software are interchangeable throughout. For example, the earlier iSBC 86/12A is compatible with the ICE 86, the in-circuit emulator for debugging hardware and software, and with the iSBC 337, the multimodule numeric data processor for high-speed mathematics.

For external interfacing, the 86/12A offers three connections: a serial port, a parallel port, and the Multibus interface. All these connections are available with at least the same user options. Since the connector pinouts stay the same, all single-board controllers are plug-compatible.

The iRMX 86 operating system standard, together with its application software, dictates that all single-board controllers have the same memory-access capability and I/O resources. To ensure compatibility between the earlier and the newer boards, the new iSBC 86/14 is designed with the same memory capabilities as the iSBC 86/12A for on- and off-board access to memory. In addition, the iSBC 86/30, with its 128 kbytes of RAM, further extends the system's capabilities. I/O controllers are contained on iSBC 86/XX boards in the same default configuration as on iSBC 86/12A boards. Thus, the software—like the hardware—is plug-compatible, virtually eliminating the need for additional software expense when upgrading a system.

If a system's RAM requirements are under 16 kbytes, an iSBC 86/05 supported by a high-speed 8-kbyte RAM Multimodule board (iSBC 302), offers better performance than either an 86/12A or any newer board. For factory applications, the 86/05 offers twice the EPROM capacity of the 86/14 or 30 board. But as RAM requirements grow past 16 kbytes, the 86/05 becomes less desirable because the additional memory required must be accessed from the Multibus. Data accesses from the bus occur at a much slower rate than those from the on-board memory.

For midrange RAM storage applications—16 to 64 kbytes—there is the iSBC 86/14 and a 32-kbyte RAM Multimodule board (iSBC 300A). The 86/30 runs as fast as the 86/14, but for RAM requirements exceeding 64 kbytes, the 86/30 continues to access its internal memory, providing faster operation. Moreover, an expanded memory space—a total of 320 kbytes consisting of 256 kbytes of RAM and 64 kbytes of EPROM—make the 86/30 the performance leader in large system applications.

intel®

ARTICLE
REPRINT

AR-267

April 1983

Choosing a Bus for Control

John Beaston
OEM Modules Operation
MULTIBUS® Architect

"Reprinted with permission from Instruments & Control Systems, March 1983 issue."

Choosing a bus for control

Which system bus is best for your application? Both business and technical factors should be considered when making a selection.

John Beaston
Multibus Architect
Intel Corporation
Hillsboro, OR 97123

Last month's installment of our "Bus boards for control" series provided a broad overview of bus-based systems, including a look at their features, advantages and disadvantages. This second installment describes the business and technical factors that must be considered when selecting a bus for your application from the dozens available.

• • • • •

The importance of selecting the best bus for your application can't be overemphasized. This is true whether you're working for an OEM or are building a one-of-a-kind product for internal use. The choice you make can impact such important areas as system performance, the engineering time needed for system configuration (which affects time-to-market for OEM's), and the ability to enhance the system in the future.

To help you with this critical decision, this article looks at both the business issues and technical aspects that should be considered when choosing a bus. It uses six of the more popular buses available to make example comparisons. These include Multibus, VME, Versabus, STD, S-100, and the Q-bus.

Business factors

Because many of the buses have similar technical capabilities, business factors often sway the final decision in bus selection. The most important of these are described in the material that follows. (Table 1 compares business factors for the six buses mentioned earlier.)

Public specification

Choosing a bus with publicly available specifications can provide some measure of compatibility among the board products supplied by different vendors. Unless all board designers use the same spec, compatibility is a hit-or-miss proposition.

Controlling body

Having a public spec, however, does not guarantee compatibility. Public specs can change with time. There-

Table 1: Business factors for various buses

	Multibus	VME	Versabus	STD	S-100	Q-bus
Public spec	Yes	Yes	Yes	Yes	Yes	Yes
Controlling body	IEEE 796	Motorola, Signetics, Mostek	Motorola IEEE P970*	IEEE P961	IEEE 696	DEC
Multiple vendors	150	21	10	75	100	10
Second sources	Yes					Yes
VLSI support	Yes	Yes	Yes			Yes
Processors	8080, 8085, 8086, 80186, 80188, 80286, Z80, Z8000, 68000, 16032, 6800, 6100, NCS800, 6802, 6808	68000, 80186, 16032	68000	8085, 8088, 6800, Z80	8080, 8085, 8088, 80186, 80286, Z80 6800, 68000, 16032	LSI-11

*Pending

fore, to prevent incompatibility and give bus users confidence, any changes made should receive an impartial review by a single controlling body.

A single point of control is critical. For example, until it was adopted by the IEEE, the S-100 bus had no control at all, let alone from a single point. Stores of incompatible S-100 boards are well-known. Now, with IEEE control and standardization, compatibility should be assured, at least for new products.

Multiple vendors

No one manufacturer can serve all the needs of the bus board marketplace. And the best bus specification in the world doesn't mean a thing if the type of board you need isn't available. In general, the number of products available on a given bus is proportional to the number of vendors supporting it. Thus, selecting a bus with strong multiple vendor support can be to your advantage.

Second sources

Some bus manufacturers are second sourcing the more popular boards. This helps assure board availability, and can also mean competitive pricing.

Processor independence

One issue that can affect future system enhancements is processor independence. Some buses are specific for a particular microprocessor family or type of processor. This can lock you into a processor family or type even though technology advancements offer better solutions. A bus which supports many processors lets you choose the best processor for your application without having to throw away your investment in the bus.

VLSI support

The final business issue is VLSI (very large scale integration) support for the bus. Some manufacturers have reduced the interface to the buses they support to a handful of VLSI devices. Having these available can greatly simplify the job of building a board, and can improve its performance and functionality by freeing up more board real estate for useful functions. It can also make off-the-shelf boards less expensive.

Technical factors

Let's turn now to a look at the technical factors in bus selection. Areas to be discussed include: address range, data width, bandwidth, interrupts, multiple bus masters, arbitration, mutual exclusion, board size, connector type, and I/O module bus. (Table 2 compares these factors for the six buses mentioned earlier.)

Address range

A system's memory requirements tend to go up as its capabilities and performance requirements increase. Therefore, you should choose a bus which supports an address space at least as large as the highest level processor you expect to use on it. This is generally a good idea even though at first, you might only be using a small portion of the address space. It will allow for later expansion.

On buses which specify expanded or optional address ranges, be very careful when selecting board products. Be sure the product decodes all of the address range you need. For example, if you use a 24-bit address range and buy a memory board which decodes only 20 bits, you'll find its address space is duplicated every 1 Mbyte on your 16 Mbyte space—which can be very inconvenient. Some buses, like VME and Versabus, have a way around this.

Data width

A processor presents a certain number of external data pins to the outside world and the bus you select should match this. While it is possible to put a 16-bit processor on a 8-bit wide bus and have the bus do two transfers each time the processor requests 16 bits, it's slow, clumsy, and complicated.

Bandwidth

Table 2 gives the maximum theoretical bandwidth for each bus at various data widths. Note that, with two exceptions, the bandwidths for any given data width are within about 20% of each other. This is because all the buses dis-

Table 2: Technical factors for various buses

	Multibus	VME	Versabus	STD	S-100	Q-bus
Address width	24	16, 24, 32	16, 24, 32	16	16 standard 24 expanded	16 standard 22 expanded
Data width	8 | 16	8 | 16 | 32	8 | 16 | 32	8	8 | 16	8 | 16
Bandwidth (Mbytes/s)	5 | 10	6 | 12 | 24	6 | 12 | 24	1	6 | 12	0.8 | 1.6
Interrupt lines	8	7	7	2	10	4
Interrupt ack	Polled	Daisy-chain	Daisy-chain	Daisy-chain	Polled	Daisy-chain
Mutual exclusion	Bus lock	RMW+bus lock	RMW+bus lock	None	Bus lock	None
Arbitration	Serial or parallel	Serial or parallel with daisy-chain	Parallel with daisy-chain	Serial	Parallel	Serial
MECHANICAL						
Form factor	6.75x12	6.3x9.2	9.25x14.5	4.5x6.5	5.1x10	5.25x8.9
Area (in.²)	81	58	134	29	51	47
Connector size	86/60	96/96	140/120	56	100	36/36
Connector type	Edge	DIN	Edge	Edge	Edge	Edge

cussed in this article use an asynchronous bus protocol with roughly the same timing. The only exceptions are the STD and Q-bus, which have relaxed timing specs.

The bandwidth figures given in Table 2 ignore bus arbitration delays and assume zero memory access times. But memory access times constitute over half the time of a typical bus transfer. They reduce the available bandwidth by more than 50%, and make the comparison come out even closer. Only the STD and Q-bus are restrained by their bus timing. The others are at the mercy of their memory boards. If you're looking for maximum performance, pick the bus that enables you to buy the fastest memory boards.

Interrupts

Systems are usually divided along functional lines. For example, a control system might be partitioned into analog I/O, digital I/O, and operator console functional modules. These modules could all be physically located on the same board and controlled by a single processor, or each module might be on a different board with individual control processors. However they're arranged, the modules must communicate.

The traditional method is with interrupts. Ideally, each module would have its own interrupt line, and each line would have a different priority, as shown in Fig. 1. Since often there are too many modules for this to be practical, bus designs are the result of a compromise. They allow the modules to share the interrupt lines, leaving it up to the interrupted device to figure out the source of the interrupt.

There are two main ways of doing this: software polling and daisy-chained acknowledge.

In software polling (Fig. 2), the interrupted module polls, or asks, each of the modules on the applicable interrupt request line. The module which generated the interrupt replies when polled.

Daisy-chained acknowledge (Fig. 3) is a hardware method. An interrupt acknowledge signal is passed down a line daisy-chained between all modules on the shared interrupt line. Modules not signalling the interrupt let the acknowledge signal pass on down the daisy-chain. Once the signal arrives at the interrupting module, the module captures it and places an identifying code on the data bus. The interupted module then reads this code to figure out the source.

It's hard to say whether one method is better than the other. Polling, being software based, is very flexible and easy to debug, but tends to be slower than a hardware method. Daisy-chaining is fast but makes the priorities of modules sharing an interrupt line depend on their positions.

If there are fewer modules than the number of interrupt lines, neither method offers an advantage.

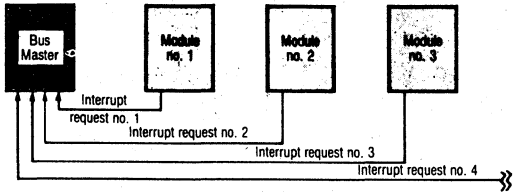
Multiple bus masters

If there is one characteristic which tells if a bus is suitable for mid-to-high end systems, it is its support of multiple bus masters. Such support greatly enhances the architectural flexibility of the bus.

Simple buses like the STD and Q-buses support only one bus master in addition to the main processor. This usually must be a direct memory access (DMA) device. Typically, both masters can't operate at the same time. If the DMA device is active, the main processor suspends operation temporarily even though it might be using only on-board private resources. Once the DMA device is done with the bus, the main processor resumes operation. This nonconcurrency generally restricts these buses to low-to-mid range performance systems.

The remaining buses support multiple bus masters in the true sense of the term. On these buses, it's possible to have many processors sharing the bus. Usually the processors operate with their on-board resources while sharing a common global pool of memory and/or I/O.

Multiple-master buses give you a lot of flexibility in system architecture. A simple system might use a single main processor board with a dumb analog I/O board. Or you could increase the system's throughput by replacing the dumb I/O board with an intelligent board containing its



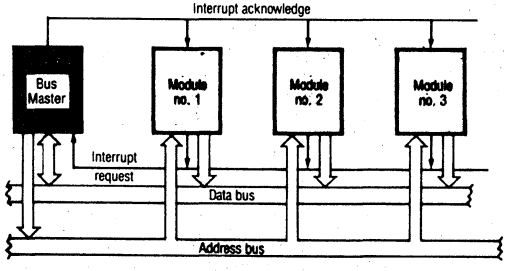
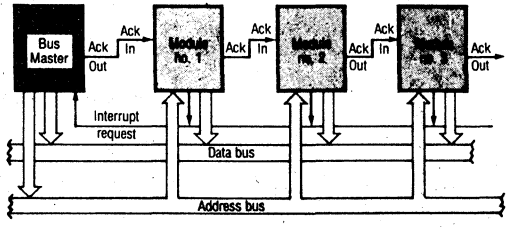


Fig. 1 (left, top): If there are more interrupt request lines than interrupting modules, the sources of the interrupts are obvious.

Fig. 2 (left, bottom): When modules must share interrupt request lines, there must be some way to find the source of an interrupt. In software polling, the bus master asks each module in turn if it is the source.

Fig. 3 (below): In daisy-chained acknowledge, the module which caused the interrupt captures the acknowledge signal to indicate that it is the source. Priority then depends upon position on the bus.



own dedicated processor. This intelligent board would operate in parallel with the main processor. The support of multiple masters lets a bus span a tremendous cost/performance range.

Arbitration

All of the buses mentioned only allow one bus master at any one time. Since several masters might want the bus at the same time, arbitration is needed to resolve the conflict and figure out which one gets the bus and which one has to wait.

This can be done with either serial or parallel methods. In parallel arbitration, each bus master (or group of masters in those buses with daisy-chained arbitration) generates a bus request, which goes to a central arbitration module. This module then resolves any simultaneous requests based upon a priority algorithm and returns a grant to use the bus to the winning master. In a purely parallel scheme, there are as many separate request and grant lines as there are masters. However, buses like VME and Versabus allow bus masters to share common request lines. The grant for each request line is then daisy-chained to determine which of the masters gets the bus.

All parallel arbitration schemes require external arbitration logic. Sometimes this logic is placed on the backplane, other times it's included on a processor board, and still other times a separate board is used. How it's done is an implementation detail that you should be aware of when you choose board products for your selected bus.

To save the cost of parallel arbitration for simpler systems, the Multibus uses a serial arbitration scheme. This requires no external logic but can only support three bus masters, which is often enough. VME can also do serial arbitration, with up to 20 masters.

Mutual exclusion

Systems which share common resources, such as a shared memory pool or a peripheral, need some way to prevent conflict. One popular method is to set a bit in memory called a semaphore to indicate that the resource is in use. A processor wanting to use the resource will read the semaphore; if it is not set, the processor will set it and use the resource.

But there can be a problem. After the bit is read, and before it can be set, another processor can read it, find it cleared, and try to set it. To avoid this problem, buses may use mutual exclusion.

The various buses provide mutual exclusion in different ways. VME and Versabus use read-modify-write (RMW) bus cycles. These are read and write cycles to the same memory location, which are treated as one bus transaction. No arbitration is allowed between the cycles so it's not possible for another processor to sneak on the bus in between read and write operations. RMW cycles are used for protecting semaphores. The semaphores, in turn, protect other shared data structures, such as pointers.

The other mutual exclusion mechanism is called bus lock. Here, when a processor wants exclusive use of a global bus resource it stops arbitration until the access is complete. The bus lock can cover more than just a read and write cycle. It could apply over many bus cycles if so desired.

It is also useful with dual-ported memory. It can inhibit access from any port of a multi-ported resource so that even though a local processor is using its local bus, it cannot violate mutual exclusion.

Board size

Perhaps the most important factor in determining the type of systems a bus supports is its form. Smaller boards are very modular. They allow you to get exactly what you need with few excess or unused functions. On the other hand, they may have lower performance. Since each function usually has its own board, and its own bus interface, more real estate goes to interfacing to the bus. Larger boards can have more functions per board, so the percentage of area devoted to the bus is lower. This leaves more room for extra functions.

The problem with larger boards is finding ones with the right mix of functions. Unused functions waste real estate, power, and money. Supporters of the larger boards (Multibus and Versabus) have recognized this problem and have developed I/O module bus structures as extensions to the primary system bus. We'll discuss these extensions shortly.

The other tradeoff on board size is performance. Many of the boards use a local bus. This is an on-board bus which the processor can use without disturbing the main system bus. These local buses tend to have very high performance, since the distances are short and arbitration is usually not required. It's good to maximize the time a processor spends on its local bus since it raises its overall performance. No matter how fast the system bus may be, the local bus is always faster.

In general, smaller boards have less room for local bus resources. This is particularly true for memory. Once the processor exceeds the amount of local memory, it must use the system bus and suffer the delays of arbitration and multiple logic gates. Larger boards have larger local memory capacity and shouldn't have to use the system bus as often; hence, their system performance can be higher. Multibus has a local bus extension (iLBX) available. This extends a board's local bus to other boards without going through the system bus.

Connector type

All of the buses in this comparison except VME use single-piece edge connectors. VME uses the European standard DIN pin-and-socket connector. Edge connectors have been around a long time and have acceptable connection reliability performance, but some people feel that pin-and-socket connectors are better in high-vibration environments. They are also gas-tight while edge connectors are not.

Another motivation for using DIN connectors is space efficiency. For connections larger than 100 pins, DIN connectors take up less board area than edge connectors. The small VME boards with fully demultiplexed 32-bit address and 32-bit data buses need DIN connectors to support all those pins.

Table 3: Comparison of I/O module buses

I/O Modular Bus	ISBX	I/O Channel
Companion system bus	Multibus	Versabus VME
Public spec	Yes	Yes
Controlling body	IEEE P959	Motorola
Multiple vendors	10	2
Address width	5	12
Data width	8 | 16	8
Bandwidth (mByte/sec)	5 | 10	2
DMA support	Yes	
I/O connection type	Daughter board	Separate cable or backplane

The other major differences are cost and current capacity. Pin-and-socket connectors are more expensive than edge connectors, and they handle less current than edge connector fingers (1to 1.5 A per pin connection vs 1 to 5 A per edge connector finger). This means that buses using pin-and-socket connectors need to use more pins to supply power.

I/O module bus

As mentioned earlier, the supporters of the larger boards have developed specialized I/O buses to improve their modularity. Multibus uses the iSBX I/O Expansion Bus and VME/Versabus have the I/O Channel. Table 3 gives a comparison of these buses.

While the goals are similar, the methods are different. The I/O Channel uses a separate ribbon cable or backplane, and can extend up to twelve feet from the host system. Up to 16 slave devices can be connected, transferring data over an 8-bit wide path.

The iSBX is a small daughter board which plugs directly into a host Multibus board, and each daughter board handles one kind of I/O. Both 8 and 16-bit data transfers are possible.

A final word

With this information you should be able to make a more informed choice of a bus for your application. Just remember that there are a lot of buses available, not just the six discussed here. And there may be more than one which will suit a particular application. ■

The author, John Beaston, will be available to answer any questions you may have about this article. Mr. Beaston can be reached at (503) 681-8080 during normal business hours.

---

intel

ARTICLE
REPRINT

AR-268

April 1983

"Standard Buses Capture Fancy of Most OEM's"

"Building Blocks for Micros"

Mitchell York
Michael Azzara
Computer Systems News

Order Number: 230727-001

Standard Buses Capture Fancy Of Most OEMs

By Mitchell York

Before OEMs or systems builders can choose a single-board computer around which to configure a multiboard system, they must first decide on a system bus.

It used to be they simply designed proprietary buses. But the evolution of board-level computers and standard bus architectures that began in the mid-1970s has changed all that: OEMs and systems houses now perceive the use of standard buses and bus-compatible board-level components as a way to reduce costs and lay the groundwork for the integration of future technological advances. The days of the proprietary bus are numbered, industry executives agree.

Several standard buses are now fighting for market share, with Intel Corp.'s Multibus leading the pack with more than half the 16-bit segment and one-third to one-half of the 8-bit market, according to managing analyst David Aronovitz of research house Gnostic Concepts Inc. Pro-Log Corp.'s STD bus is another major contender in the 8-bit arena, he said.

And both are bracing for stiff competition from a more recent offering sponsored by a coalition of semiconductor manufacturers led by Motorola Inc. The coalition's VME bus is compatible with 8-, 16-, and 32-bit architectures.

Additional competition is provided in the 8-bit market by the S-100 bus developed by MITS Inc. and in the 16-bit market by Digital Equipment Corp.'s Q-Bus. The proprietary bus is still an option, but industry experts are betting against it.

Many OEMs and systems builders are looking to standard buses to reduce engineering overhead and help position them to expand and modify their systems when new technologies become available. Those two factors — economics and technological advances — are the driving forces behind the mounting attention being focused on standard buses, according to industry executives.

"The technology has moved to a point where the user can integrate the baseline of his system onto a single-board computer," said Gary Sawyer, board products marketing manager for Intel's OEM Modules Group. "But we have very few customers out there who use a single-board solution. The primary advantage (of a single-board computer) is more content on a single board, so that as an OEM configures a system on his bus, he has a simpler configuration to perform. And that's why the bus question becomes so important," Sawyer said.

"The market, now that it has reached an adolescence, is maturing to a point where customers have to consider what bus they're deciding on first, and where that bus can take them in the future. Then they can evaluate the vendors and the products that are available.

While many OEMs still stand by proprietary buses, there are growing indications that the economics of the OEM business will not allow them to resist standardization for long. "There are still companies out there that are emphatically clinging to dedicated systems architectures for mini and microcomputers," said Ray Burkley, president of Astraea Computer Corp., Sunnyvale, Calif., which is marketing a VME bus-compatible single-board computer. "They are the ones that will end up getting hurt in the long run, because all the other reasons germane to using (standard) buses make sense."

"I think a proprietary bus is a definite tactical error," said Gnostic's Aronovitz. "It locks you out of the market, it forces you to design everything in-house, and it delays your entry into the marketplace, he said.

Standard Advantage

One reason for moving to standard architecture is that it is becoming prohibitive to design systems starting from the chip level, some OEMs said, unless the number of systems needed is high and unless the technology is likely to remain constant for a long time.

Observers say there is a proliferation of systems builders making small quantities of systems aimed at profitable vertical markets. Because these OEMs do not enjoy economies of scale and cannot afford to build from scratch, they are increasingly turning to standard architectures. In addition, with semiconductor and microprocessor technology moving so rapidly, it is likely that systems will have increasingly shorter marketing life cycles, thus further fueling the standard bus movement.

If systems builders are thinking about standard buses, vendors cannot be far behind. Aronovitz reports that there are 300 vendors in the board-level market and all have aligned their products with one or more of the standard buses.

The attention standard buses are receiving cannot help but grow exponentially in the coming years, industry executives say, because of backing from such powerhouse companies as Intel, DEC, and Motorola. Those companies and several others are helping create vast secondary markets to supply bus-compatible products geared for the systems builder. And the more products put out on the market for OEMs, the more OEMs are likely to use them and the buses they are made for, according to observers.

While OEMs and their suppliers may differ about which bus is best, there seems to be widespread agreement on the need to seriously address systems building starting at the board level and employing some standardized bus. It's a simple question of speed, according to Jeff Gorin, product manager of Motorola's MOS Integrated Circuits Group.

Faster Integration

"The fundamental advantage of modular product implementations using boards with common bus interfaces is that the systems integrator is going to be able to accomplish system integration faster — assuming he can find board-level components — by plugging together existing, documented boards rather than starting from scratch from chips," said Gorin.

Speed of product entry was the key factor that led Control Automation Inc., a Princeton, N.J.-based maker of factory automation equipment, to adopt an industry-standard bus. The company, which makes industrial robots and vision processors, was concerned with cutting down its engineering time and ramping up production, said director of electronic development Abe Abramovich.

Control Automation accordingly decided to build its robots with boards compatible with Multibus. "We came up to speed in our product development cycle more rapidly by buying off-the-shelf computer boards with all the functionality we require. By not having to design circuitry, we were able to apply our energy where it was really required — value-added interfaces."

The time savings that standard buses yield can be enormous, said Astraea's Burkley. Standard buses, with the myriad off-the-shelf board-level products available, allow an OEM to get to market from six to 18 months sooner than if the systems builder were to design his own proprietary bus and boards, Burkley said. "It's a make-buy decision. Do you want to take 30 months to develop a proprietary product, or buy at the board level, add software, and be in the marketplace?"

Once the decision is made to build with standard boards that are compatible with standard buses, the issue becomes which bus to use. Several have emerged as leaders because they

have won acceptance from components manufacturers who intend to create an array of products for systems builders.

One of the latest buses to achieve a growing degree of recognition is the VME bus. Motorola, Signetics Corp., Mostek Corp. and Thomson-EFCIS last year all threw their weight behind VME, and now an IEEE committee chaired by Signetics' engineering manager Michael Clader has been formed to draft a standard.

Proponents of VME, which is specially designed for use with the 68000 chip, say its main attribute is its ability to support 8-, 16-, and 32-bit microprocessors. This is important to OEMs that want to design 16-bit systems now that will work in the 32-bit environment when the technology is available.

"The VME bus is the only bus architecture that really allows us to build things today for both 16- and 32-bit," Burkley said. "It's our intention to build 32-bit supermini-computers. A lot of LSI components we need aren't there today, so we selected a bus architecture that will allow us to grow from the present 16-bit capability to 32-bit.

"We'll start building board-level products now that we can sell and which will generate feedback as well as profits. When full 32-bit capability is available, we'll only have to add a couple of boards to do the upgrade."

VME is also cited by systems integrators as more expandable than several other major bus options. "It gave me more real estate. That was critical to me," said Roger Vass, president of Victory Computer Systems Inc., San Jose, Calif. With VME, he said, Victory can achieve the cost efficiency of single-board computer design without sacrificing system expandability.

"VME has architectural aspects that are directly coincident with my marketing strategy," Vass continued, "I am going into the fault-tolerant, multiple-CPU, transaction processing market within a year." VME's speed is essential to succeeding in Victory's targeted markets, Vass said. "I need that kind of bus speed in order to execute fault-tolerant systems. I got it."

What Victory also got by adopting VME, Vass said, is a bus architecture that has been endorsed by major semiconductor houses, which gives it market credibility. "And it's a new bus. From a marketing standpoint, it's attractive because it's new."

VME is frequently compared to Intel's well-established Multibus. VME supporters maintain that Multibus can not fully support microprocessors as powerful as the 68000. Multibus enthusiasts counter by noting the constant stream of upgrades that have been made in the bus since its introduction to the OEM market in 1976.

Sawyer noted that even though Multibus was originally conceived as a bus that would address the 8-bit market, it has been adapted for 16-bit. "There have been six improvements made in Multibus to adapt to new VLSI. We are at the forefront of yet another evolution of Multibus to adapt it to our 80286 microprocessor."

LBX Refinement

The latest Multibus refinement is LBX — Local Bus Extension — which the company said allows a microprocessor to address up to 16 Mbytes of local system memory at very high speeds. LBX, Intel said, supports a data transfer rate of 9.5 Mbytes per second for 8-bit data and 19 Mbytes per second for 16-bit data. Intel said six single-board computer manufacturers have notified them of plans to develop products based on LBX.

It is just this kind of secondary market support that is crucial to the success of a bus, according to industry executives. There are now 165 suppliers of more than 1000 Multibus-compatible products, according to Fred Mazenac, president of Ironoak Co., the La Jolla, Calif.-based publisher of the *Multibus Buyers Guide*.

Motivating Factor

"Very clearly, one of the most important motivating factors in users selecting a bus is the number of vendors and products available for that particular bus," said Mazenac, who is now involved in assembling a Multibus manufacturers association.

Intel said the market last year for single-board computers was $375 million, about one-half of which was Multibus-compatible. By 1985, Intel said, revenues are expected to grow to $800 million, and Multibus is expected to retain its approximate 50 percent market share.

Sawyer said another boon to Multibus is its endorsement by the IEEE, which recently declared it a standard. With a firm, unchangeable standard in place, more vendors are expected to develop Multibus-compatible products, Sawyer said.

While Multibus is being used for a broad range of applications, Pro-Log's STD bus is geared primarily for industrial uses. Companies that build products compatible with the STD bus, also known as Standard bus, note that its small card size makes it attractive for single-board systems vendors that focus on industrial applications.

Dick Thomas, director for product development for Pro-Log, said STD "has an I/O intensity rather than a number-crunching intensity," which makes it better for control purposes than for data processing.

Thomas said the main advantage of STD is its price competitiveness. STD-based products are less expensive than Multibus-based products, but also less versatile. "What tends to happen (with the large Multibus card) is that people use it up and put a lot of things on the board just because there is a lot of room. It can be overkill.

"If you need what's on a Multibus card, like two I/O channels, it probably becomes more cost-effective. But that rarely happens, and that's where our advantage is. STD allows you to modularize to specific functions. Consequently there is a big price advantage," Thomas said.

While STD hasn't made its way into many office systems because of its data processing limitations, the older S-100 bus is most often used in small business systems. S-100, developed in 1975 for the Altair home computer, "is very flexible and modular," said Mark Garetz, president of CompuPro Systems, Oakland Airport, Calif.

CompuPro works with S-100, Garetz said, because it can be used with almost any processor. "It's not that hard to put various processors on it; it's not that processor specific."

Garetz said the cost/performance ratio of the S-100 "is unequaled by any of the other bus structures." Equivalent products based on Multibus can cost three to four times as much as S-100 products, he said. □

Building Blocks For Micros

As Systems Get Smaller, The Role Of Single-Board Computers Gets Bigger

By Michael Azzara

On the most basic level, a single-board computer is just what the name implies: the essential components of a computer system — a CPU, memory, and at least one I/O channel — on a single printed-circuit board. But the widespread assumption that it's a single board that constitutes a computer system is usually wrong.

So what's right? Single-board computer vendors agree that their products are malleable; they can be used alone in some applications, while in others they must be tied with additional boards to make a working system.

The fact is that use of single-board computers is pervasive in the microcomputer industry: Such boards are the building blocks of most modern microcomputer systems. The majority are manufactured from scratch by high-volume general-purpose hardware vendors, but more and more systems houses and OEMs in narrower vertical markets are buying single-board computers, usually from one of the approximately half-dozen semiconductor houses that analysts estimate control about 98 percent of the board-level products market.

The phrase single-board computer was coined by Intel Corp. in 1976 and has become virtually a generic term for microprocessor-based CPU boards. Many single-board computer vendors, however, point out that the term is somewhat misleading, since most of their OEMs actually include more than one board in their final products. But they agree nonetheless that there are important advantages to cramming the essential components of a system onto a single board rather than using three, four, or more boards strung together to make a computer system.

The primary benefits they cite are increases in performance and decreases in both cost and physical size, not necessarily in that order — and not all vendors agree on the relative importance of each.

On the flip side of the coin, single-board computers are a cost-effective solution for OEMs in only a narrow range of volumes. In very low-volume applications, an OEM may find a packaged system to be a better buy, while high-volume applications may call for an OEM to turn to component-level integration. Most of the applications that fall in that range are technical in nature, such as controlling scientific instruments or industrial machines.

Within that volume range — usually anywhere up to 1000 units during the life of a system — the advantages far outweigh the disadvantages, vendors say.

Performance is improved over multiboard systems because execution speed and reliability both increase when more work is performed within the confines of one board, rather than through communications on a system bus. Cost goes down because using one board to perform functions that once required three or more saves on board expenses. And because the final system can be made smaller than multiboard systems, doors are opened to applications that previously required smaller systems than could be built.

Besides those technical advantages, many vendors point out that buying single-board computers allows OEMs to concentrate their efforts on their own expertise in a particular applications area, rather than expending engineering talent to duplicate electronics that are already available in standard packages.

There are also the obvious benefits derived by the OEMs whose applications can be handled by one stand-alone single-board computer. These applications tend to be in the area of industrial automation and process control, such as the control of medical and scientific instruments.

All single-board computers, however, are not sold to OEMs. In fact, the market in 1982 was evenly split between the captive and OEM segments, according to managing analyst David Aronovitz of Gnostic Concepts Inc. He said, however, that he expects the captive percentage to grow as board manufacturers like Intel and Mostek Corp. begin selling more of their own systems. Adding to the captive segment are companies like NCR Corp., which makes boards for itself but doesn't sell in the OEM market.

Aronovitz said Intel shipped approximately 30,000 16-bit single-board computers in 1982 — about one-third to one-half of the 16-bit market. Specific figures for 8-bit single-board computers weren't available, but Aronovitz said volumes are much higher — in the hundreds of thousands.

In the OEM segment, most single-board computers are sold through the major electronics distributors such as Avnet Inc. and Schweber Electronics Inc. Smaller vendors sell through manufacturers representatives, Aronovitz said.

But despite their increasing popularity, single-board computers aren't a panacea. Before buying such a board, OEMs must consider several factors, primarily the questions of application and volume. OEMs must first consider whether the application is suited to a single-board solution. If it is, the OEM must then determine whether the solution promises to sell in volumes that fall into the range that makes board-level integration cost-effective.

How long it will take to bring a product to market must also be taken into account. If an OEM must react quickly to a closing market window, single-board computers provide a faster answer than in-house design, and buying packaged systems is a solution that's faster still.

"There are economies of scale at the various levels of integration," said Gary Sawyer, board products marketing manager for Intel's OEM Modules Operation. "Components are the lowest cost, highest risk, and take the longest time to (bring a product to) market. At the other end are systems where the OEM's value-added is software. They're the highest cost, lowest risk, and take the least time to market.

"At the board level, we're right in the middle," Sawyer said. "Primarily, we're talking about a board crammed with technology, about 81 inches square, that goes into a laboratory or factory environment," he said. About 80 percent of Intel's single-board computers are used in those types of technical applications, while applications for the remainder are varied.

Sawyer said microprocessors are pervasive in the desktop computer area, but because the volumes for those applications are enormous, manufacturers primarily use components and design their own boards.

More Than One

Sawyer admitted that despite the name single-board computer, most OEMs use more than one board.

"But there is a great benefit to solving your problems within the confines of a single board and using your bus as a system resource. You move questions and answers across the system, on the bus, but keep real hard work off the bus. When you have hundreds of thousands of bytes of memory in a single-board computer and can do the work on such a localized basis, your performance benefits are enormous," Sawyer said.

"Another thing is that with several boards, you have to continually go back and forth through the bus to fetch instructions, move

memory, etc. What you want to do, in my view, is localize those functions and maintain that precious system bus activity for system-kind of functions, like communications between the CPU and disk drives or CRTs. You don't want to use that bus to move bits and bytes around like that," concluded Sawyer.

Even as single-board computer vendors tout their latest semiconductor advances, waiting offstage is the next major step in the miniaturization process — single-chip computers.

Semiconductor houses are already selling what they call microcontrollers — chips that include not only a microprocessor, but nonvolatile memory and I/O capability as well — in large volumes for simple control applications. Industry observers believe it's just a matter of time before chip makers are able to squeeze enough sophistication into a silicon wafer to make it indistinguishable from current single-board computers.

"That's the new wave," said Rod Zwonitzer, system products marketing manager for Mostek Corp. "Taking what you have on a single board and shrinking it onto a chip. This technology is a continually shrinking thing."

Existing microcontrollers, with most software embedded in ROM, are used to control predetermined functions in peripheral devices such as disk drives, CRTs, printers and keyboards, according to Graham Alcott, Intel Corp.'s microcontroller products marketing manager.

They are also used in much higher volumes in consumer products such as televisions, telephones, automobiles and microwave ovens.

Extremely Small

Because of their extremely small size and price, microcontrollers can cost-effectively replace mechanical and electromechanical devices in those applications, said Joseph Baranowski, assistant marketing manager of Intel's microcontroller operation.

And because those are such high-volume applications, microcontroller production is high. According to a recent Dataquest Inc. market report, U.S. semiconductor houses shipped approximately 50 million microcontroller chips in the third quarter of 1982, Baranowski said. He added that Intel will not even entertain orders for fewer than 1000 pieces.

Intel's Latest

Intel's latest device, the 8096, packs in a 16-bit CPU, 8k bytes of ROM, 256 bytes of RAM, and multiple I/O channels, including eight analog-to-digital converters, he said.

The analog communications capability is an important aspect of microcontrollers, since it enables them to control the functions of devices in a realtime environment, Alcott said.

Compared with Intel's first single-board computer, the 8010, which was based on the 8-bit 8080 microprocessor chip, the 8096 single-chip computer provides double the memory and more than double the performance, according to Baranowski.

"A lot of what we're now doing on single chips were actually board-level products less than five years ago," he said.

Future Trend

"The trend in the future is that microcontrollers will continue to be a dominant way people design," said Baranowski.

"The typical office of the future will have numerous disks, keyboards, printers, all with one or more microcontrollers, and there will likely be only one microprocessor acting as CPU for the whole system," he said.

Besides Intel and Mostek, prominent microcontroller manufacturers in the United States include Motorola Inc. and Zilog Inc.

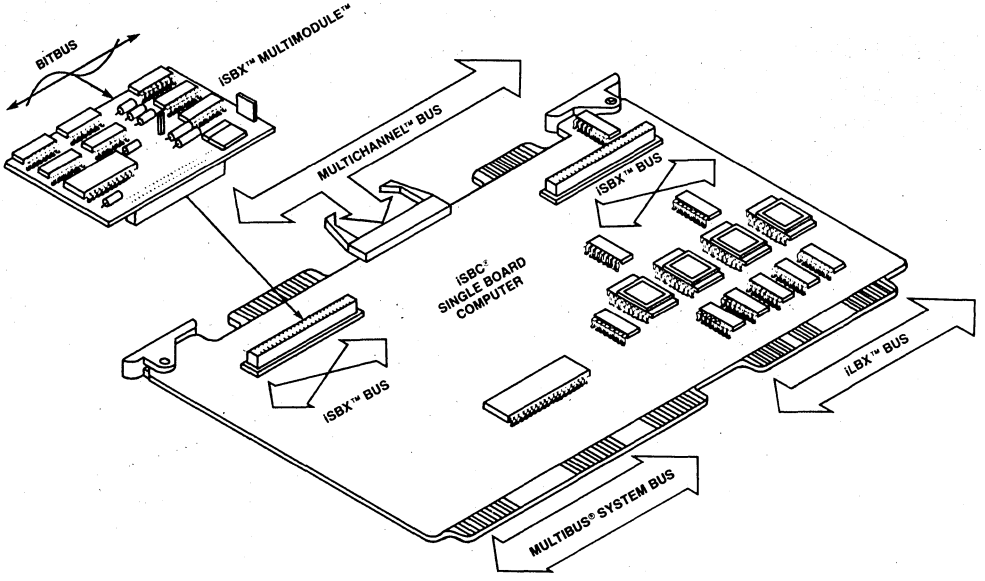
MULTIBUS® I Architecture 3

PRELIMINARY

MULTIBUS® SYSTEM BUS

- **IEEE 796 industry standard system bus**
- **Supports multiple processor systems with multi-master bus structure**
- **8-bit and 16-bit devices share the same MULTIBUS® system resources**
- **Foundation of Intel's Total System Architecture: MULTIBUS,® iLBX,™ MULTICHANNEL™, BITBUS™ and iSBX™ buses**
- **16 Mbyte addressing capability**
- **Bus bandwidth of up to 10 megabytes per second**
- **Supported by a complete family of single board computers, memory, digital and analog I/O, peripheral controllers, graphics and speech recognition, packaging and software**
- **Supported by over 150 vendors providing over 1000 compatible products**

The MULTIBUS® System bus is one of a family of standard bus structures resident within Intel's total system architecture. The MULTIBUS interface is a general purpose system bus structure containing all the necessary signal lines to allow various system components to interact with one another. This device interaction is built upon the master-slave concept. The "handshaking" between master and slave devices allows modules of different speeds to use the MULTIBUS interface and allows data rates of up to 5 million transfers per second. The MULTIBUS system bus can support multiple master devices (16) on a 18 inch backplane and can directly address up to 16 megabytes of memory. As a non-proprietary, standard system bus, the MULTIBUS interface has become the most prominent 8/16-bit microcomputer system bus in the industry with over 150 vendors supplying over 1000 MULTIBUS compatible products. Its success as the industry standard has been reinforced by adoption of the MULTIBUS specification by the Institute of Electrical and Electronic Engineers — (IEEE 796 System Backplane Bus). MULTIBUS-based systems have been designed into applications, such as, industrial automation and control, office systems and word processing, graphics systems and CAD/CAM, telecommunications systems and distributed processing.



FUNCTIONAL DESCRIPTION

Architectural Overview

The MULTIBUS® system bus is the physical framework and the conceptual foundation of Intel's total system architecture. It is a general purpose system bus used in conjunction with the single board computer concept to provide a flexible mechanism for inter-module processing, control and communication. The MULTIBUS interface supports modular CPU, memory and I/O expansion in flexible, cost effective microcomputer system configurations. These configurations implement single board computers and expansion modules in a multiple processor approach to enhance system performance. This enhanced performance is achieved through partitioning of overall system functions into tasks that each of several processors can handle individually. When new system functions are added (peripherals) more processing power can be applied to handle them without impacting existing processor tasks.

Structural Features

The MULTIBUS interface is an asynchronous, multiprocessing system bus designed to perform 8-bit and 16-bit transfers between single board computers, memory and I/O expansion boards. Its interface structure consists of 24 address lines, 16 data lines, 12 control lines, 9 interrupt lines, and 6 bus exchange lines. These signal lines are implemented on single board computers and a mating backplane in the form of two edge connectors resident on 6.75″ x 12.00″ form factor PC boards. The primary 86-pin P1 connector contains all MULTIBUS signal lines except the four address extension lines. The auxiliary 60-pin P2 connector contains the four MULTIBUS address extension lines, and reserves the remaining 56 pins for implementing the iLBX™ Execution Bus into the MULTIBUS system architecture.

Bus Elements

The MULTIBUS system bus supports three device categories: 1) Master, 2) Slave, 3) Intelligent Slave.

A bus master device is any module which has the ability to control the bus. This ability is not limited to only one master device. The MULTIBUS interface is capable of supporting multiple masters on the same system through bus exchange logic. Once access has been acquired by a master device, it has a period of exclusive control to affect data transfers through a generation of command signals, address signals and memory or I/O addresses.

A bus slave device is a module that decodes the address lines on the MULTIBUS and acts upon the command signals from the bus masters. Slave devices are not capable of controlling the MULTIBUS interface.

The intelligent slave has the same bus interface attributes as the slave device but also incorporates an on-

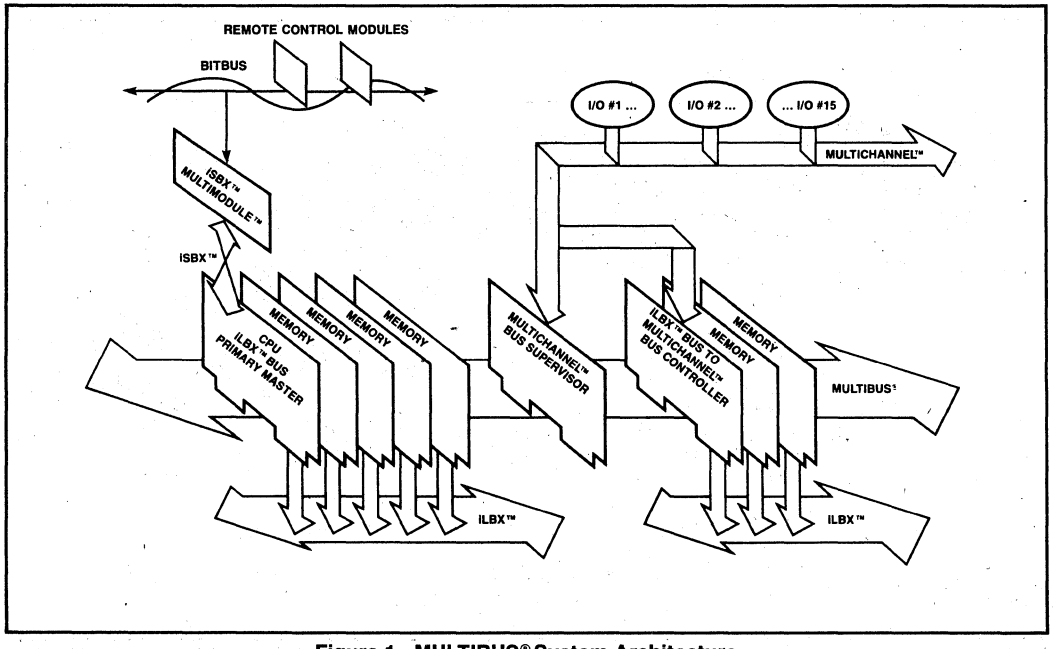


Figure 1. MULTIBUS® System Architecture

board microprocessor to control on-board memory and I/O tasks. This combination of on-board processor, memory and I/O allow the intelligent slave to complete on-board operations without MULTIBUS access.

Bus Interface/Signal Line Descriptions

The MULTIBUS system bus signal lines are grouped into five classes based on the functions they perform: 1) control lines, 2) address and inhibit lines, 3) data lines, 4) interrupt lines, 5) bus exchange lines. Figure 2 shows the implementation of these signal lines.

The MULTIBUS control lines are broken down into five sub-groups: clock signals (2), commands (4), acknowledge (1), initialize (1), and lock (1). The two clock signals provide for the generation of a master clock for the system and the synchronization of bus arbitration logic. The four command lines are the communications links between the bus masters and bus slaves, specifying types of operations to be performed such as reads or writes from memory or I/O. The transfer acknowledge line is the slave's acknowledgement that a requested action of the master is complete. The initialize signal is generated to reset the entire system to a known state. The lock signal is used by an active bus master to lock dual-ported for mutual exclusion.

The address and inhibit lines are made up of 24 address lines, two inhibit lines, and one byte control line. The 24 address lines are signal lines used to carry the address of the memory location or the I/O device that is being referenced. These 24 lines allow a maximum of 16 million bytes of memory to be accessed. When addressing an I/O device, sixteen address lines are used to address a maximum of 64 thousand devices. The two inhibit lines are used to allow different types of memory (RAM, ROM, etc.) having the same memory address to be accessed in a preferred priority arrangement. The byte control line is used to select the upper byte of a 16-bit word in systems incorporating 16-bit memory and I/O modules.

The MULTIBUS interface supports sixteen bi-directional data lines to transmit or receive information to or from a memory location or an I/O port.

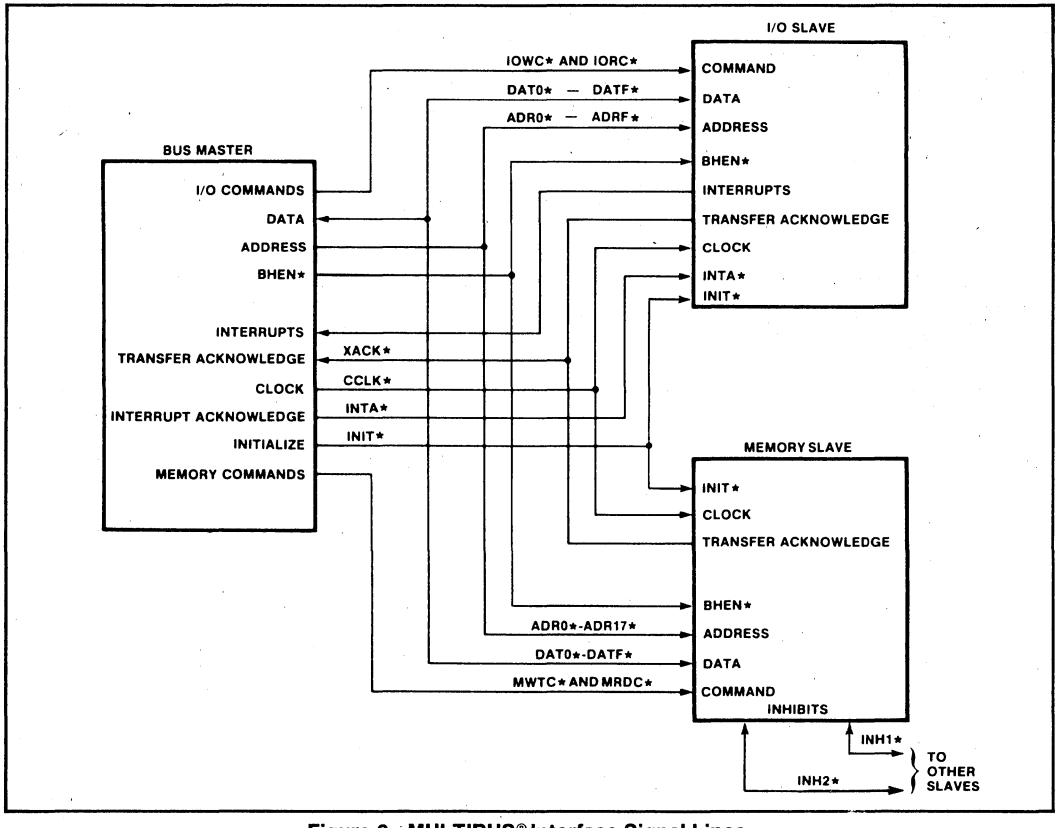


Figure 2. MULTIBUS® Interface Signal Lines

The MULTIBUS interrupt lines consist of eight interrupt request lines and one interrupt acknowledge line. Interrupts are requested by activating one of the eight interrupt request lines. The interrupt acknowledge signal is generated by the bus master when an interrupt request is received. It effectively freezes interrupt status and requests the placement of the interrupt vector address onto the data lines. There are six bus exchange lines that support two bus arbitration schemes on the MULTIBUS system bus. A bus master gains control of the bus through the manipulation of these signals. The bus request, bus priority, bus busy, and bus clock signals provide for a slot dependent priority scheme to resolve bus master contention on the MULTIBUS interface. Use of the common bus request signal line can save arbitration time by providing for a higher priority path to gain control of the system bus.

Bus Operation Protocol

DATA TRANSFER OPERATION

The data transfer operation of the MULTIBUS system bus is a straight-forward implementation of an asynchronous master-slave handshaking protocol. Figures 3 and 4 show the basic timing for a read and write data transfer operation. A MULTIBUS data transfer begins by having the bus master place the memory or I/O port address on the address bus. If the operation is a write, the data is also placed on the data lines at this time. The bus master then generates a command (I/O read or write, or memory read or write) which activates the appropriate bus slave. The slave accepts the data if it is a write operation, or places data on the data bus if it is a read. A transfer acknowledge is then sent to the bus master by the bus slave, allowing the bus master to complete its cycle, removing the command from the command line, and then removing the address and data from the MULTIBUS interface.

INTERRUPT OPERATIONS

The MULTIBUS interface supports two types of interrupt implementation schemes, Non-Bus Vectored and Bus Vectored. Non-Bus vectored interrupts are interrupts handled on the bus master which do not require the MULTIBUS interface for transfer of the interrupt vector address. The interrupt vector address is generated by the interrupt controller on the master and transfered to the processor over the local bus when an interrupt request line is activated by a slave module over the MULTIBUS interface. Bus vectored interrupts are interrupts which transfer the interrupt vector address along the MULTIBUS data lines from the slave to the bus master using the interrupt acknowledge command signal for synchronization. When an interrupt request occurs, the interrupt control logic on the bus master interrupts the processor, generating an interrupt acknowledge command that freezes the interrupt logic on the bus for priority resolution and locks the MULTIBUS system bus. After the bus master selects the highest priority active interrupt request lines, a set of interrupt sequences allow the bus slave to put its interrupt vector address on the data lines. This address is used as a pointer to interrupt the service routine.

BUS EXCHANGE TECHNIQUES

The MULTIBUS system bus can accommodate several bus masters on the same system, each one taking control of the bus as it needs to affect data transfers. The bus masters request bus control through a bus exchange sequence.

The MULTIBUS interface provides for two bus exchange priority techniques: a serial technique and a parallel technique. In a serially arbitrated MULTIBUS system, requests for system bus access are ordered by priority on the basis of bus slot location. Each master on the bus notifies the next lower priority master when it needs to use the bus, and it monitors the bus request status of the next higher priority-master. Thus, the masters pass bus requests along from one to the next in a daisy chain fashion. The parallel bus arbitration technique resolves system bus master priorities using external hardware in the form of a priority resolution circuit. This parallel arbitration logic is included in many commercially available cardcages.

Mechanical Implementation

BUS PIN ASSIGNMENTS

Printed circuit boards (6.75″ × 12.00″) designed to interface to the MULTIBUS system bus have two connectors which plug into the bus backplane. These connectors, the 86-pin P1 (Primary) and the 60-pin P2 (Auxiliary), have specific pin/signal assignments. Because of this, the designer must insure that the MULTIBUS backplane being designed is compatible (pin-for-pin) with these two connectors. Tables 1 and 2 show the pin/signal assignments for the P1 and P2 edge connectors. The MULTIBUS interface connection is accomplished via a rigid backplane that has connectors that mate to the P1 (43/86-pin) board edge connector and allows for connectors that mate to the P2 (30/60-pin) board edge connector. Figure 5 shows a typical MULTIBUS backplane. Figure 6 displays the connector and pin numbering convention. Figure 7 shows the standard MULTIBUS form-factor printed wiring board outline.

Please refer to Intel's MULTIBUS specification and iLBX bus specification for more detailed information.

Table 1. MULTIBUS® Pin/Signal Assignment — (P1)

	Pin	(Component Side)		Pin	(Circuit Side)	
		Mnemonic	Description		Mnemonic	Description
Power Supplies	1	GND	Signal GND	2	GND	Signal GND
	3	+5V	+5Vdc	4	+5V	+5Vdc
	5	+5V	+5Vdc	6	+5V	+5Vdc
	7	+12V	+12Vdc	8	+12V	+12Vdc
	9		Reserved, bussed	10		Reserved, bussed
	11	GND	Signal GND	12	GND	Signal GND
Bus Controls	13	BCLK*	Bus Clock	14	INIT*	Initialize
	15	BPRN*	Bus Pri. In	16	BPRO*	Bus Pri. Out
	17	BUSY*	Bus Busy	18	BREQ*	Bus Request
	19	MRDC*	Mem Read Cmd	20	MWTC*	Mem Write Cmd
	21	IORC*	I/O Read Cmd	22	IOWC*	I/O Write Cmd
	23	XACK*	XFER Acknowledge	24	INH1*	Inhibit 1 (disable RAM)
Bus Controls and Address	25	LOCK*	Lock	26	INH2*	Inhibit 2 (disable PROM or ROM)
	27	BHEN*	Byte High Enable	28	AD10*	
	29	CBRQ*	Common Bus Request	30	AD11*	Address
	31	CCLK*	Constant Clk	32	AD12*	Bus
	33	INTA*	Intr Acknowledge	34	AD13*	
Interrupts	35	INT6*	Parallel	36	INT7*	Parallel
	37	INT4*	Interrupt	38	INT5*	Interrupt
	39	INT2*	Requests	40	INT3*	Requests
	41	INT0*		42	INT1*	
Address	43	ADRE*		44	ADRF*	
	45	ADRC*		46	ADRD*	
	47	ADRA*	Address	48	ADRB*	Address
	49	ADR8*	Bus	50	ADR9*	Bus
	51	ADR6*		52	ADR7*	
	53	ADR4*		54	ADR5*	
	55	ADR2*		56	ADR3*	
	57	ADR0*		58	ADR1*	
Data	59	DATE*		60	DATF*	
	61	DATC*		62	DATD*	
	63	DATA*	Data	64	DATB*	Data
	65	DAT8*	Bus	66	DAT9*	Bus
	67	DAT6*		68	DAT7*	
	69	DAT4*		70	DAT5*	
	71	DAT2*		72	DAT3*	
	73	DAT0*		74	DAT1*	
Power Supplies	75	GND	Signal GND	76	GND	Signal GND
	77		Reserved, bussed	78		Reserved, bussed
	79	−12V	−12Vdc	80	−12V	−12Vdc
	81	+5V	+5Vdc	82	+5V	+5Vdc
	83	+5V	+5Vdc	84	+5V	+5Vdc
	85	GND	Signal GND	86	GND	Signal GND

All Reserved pins are reserved for future use and should not be used if upwards compatibility is desired.

*Note: The Reserved MULTIBUS P2 connector pin/signal assignments are contained in Intel's iLBX Bus Specification.

Table 2. MULTIBUS® Pin/Signal Assignment — (P2)

	Pin	(Component Side)		Pin	(Circuit Side)	
		Mnemonic	Description		Mnemonic	Description
	1		Reserved	2		Reserved
	3		Reserved	4		Reserved
	5		Reserved	6		Reserved
	7		Reserved	8		Reserved
	9		Reserved	10		Reserved
	11		Reserved	12		Reserved
	13		Reserved	14		Reserved
	15		Reserved	16		Reserved
	17		Reserved	18		Reserved
	19		Reserved	20		Reserved
	21		Reserved	22		Reserved
	23		Reserved	24		Reserved
	25		Reserved	26		Reserved
	27		Reserved	28		Reserved
	29		Reserved	30		Reserved
	31		Reserved	32		Reserved
	33		Reserved	34		Reserved
	35		Reserved	36		Reserved
	37		Reserved	38		Reserved
	39		Reserved	40		Reserved
	41		Reserved	42		Reserved
	43		Reserved	44		Reserved
	45		Reserved	46		Reserved
	47		Reserved	48		Reserved
	49		Reserved	50		Reserved
	51		Reserved	52		Reserved
	53		Reserved	54		Reserved
Address	55	ADR16*	Address Bus	56	ADR17*	Address Bus
	57	ADR14*		58	ADR15*	
	59		Reserved, Bussed	60		Reserved, Bussed

All Reserved Pins are reserved for future use and should not be used if upwards compatibility is desired.

*Note: The Reserved MULTIBUS P2 connector pin/signal assignments are contained in Intel's iLBX Bus Specification.

SPECIFICATION

Word Size

Data — 8 and 16-bit

Memory Addressing

24-bits — 16 megabyte – direct access

I/O Addressing

16-bits — 64 Kbytes

Maximum Bus Backplane Length

18 inches

Bus Devices Supported

16 total devices — (Master, Slave, Intelligent Slave)

Bus Bandwidth

10 megabytes/sec — 16-bit

5 megabytes/sec — 8-bit

Bus Exchange Cycle

200 nsec – Best Case; 300 nsec – Worst Case (assuming no bus master is currently active on the bus.)

Electrical Characteristics

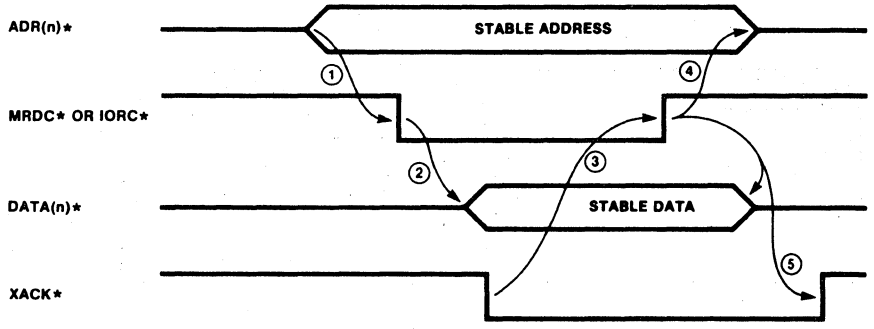
BUS POWER SUPPLY SPECIFICATIONS

Table 3.

	Standard[1]			
Parameter	Ground	+5	+12	−12
Mnemonic	GND	+5V	+12V	−12V
Bus Pins	P1-1,2,11,12, 75,76,85,86	P1-3,4,5,6, 81,82,83, 84	P1-7,8	P1-79,80
Tolerance	Ref.	±1%	±1%	±1%
Combined Line & Load Reg	Ref.	0.1%	0.1%	0.1%
Ripple (Peak to Peak)	Ref.	50 mV	50 mV	50 mV
Transient Response (50% Load Change)		100 μs	100 μs	100 μs

[1]Point of measurement is at connection point between motherboard and power supply. At any card edge connector a degradation of 2% maximum (e.g. voltage tolerance ±2%) is allowed.

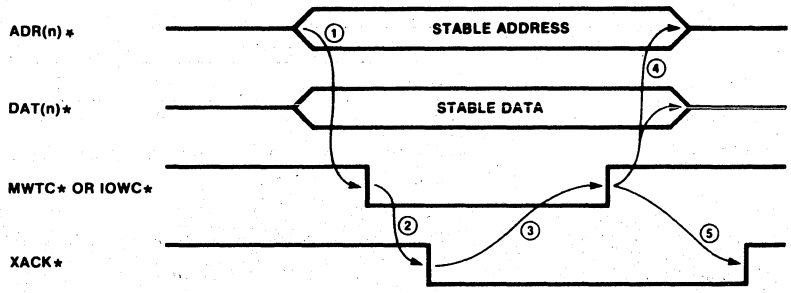
BUS TIMING



1. ADDRESS SETUP TIME: 50 NANOSECONDS MINIMUM.
2. TIME REQUIRED FOR SLAVE TO GET DATA ONTO BUS IN ACCORDANCE WITH SETUP TIME REQUIREMENT. XACK* CAN BE ASSERTED AS SOON AS DATA IS ON BUS.
3. TIME REQUIRED FOR MASTER TO REMOVE COMMAND.
4. ADDRESS AND DATA HOLD TIME; 50 NANOSECONDS MINIMUM.
5. XACK* AND DATA MUST BE REMOVED FROM THE BUS A MAXIMUM OF 65 NANOSECONDS AFTER THE COMMAND IS REMOVED.

Figure 3. Memory or I/O Read Timing

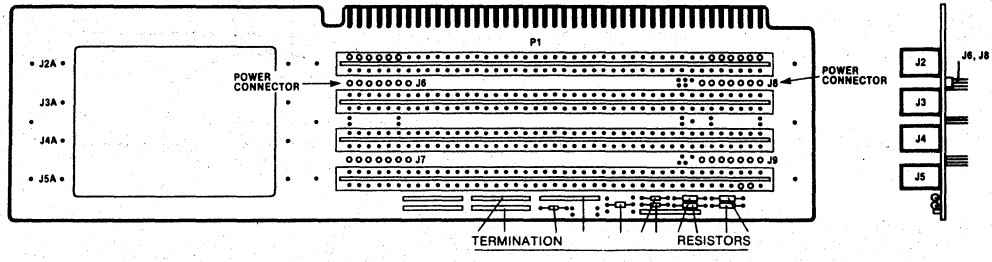
BUS TIMING (Con't)



1 ADDRESS AND DATA SETUP; 50 NANOSECONDS MINIMUM.
2 TIME REQUIRED FOR SLAVE TO ACCEPT DATA.
3 TIME REQUIRED FOR MASTER TO REMOVE COMMAND FROM BUS.
4 ADDRESS AND DATA HOLD TIME; 50 NANOSECONDS MINIMUM.
5 XACK* MUST BE OFF THE BUS 65 NANOSECONDS AFTER COMMAND.

Figure 4. Memory or I/O Write Timing

Physical Characteristics



PARTS LIST

1 PWB TERMINATION BACKPLANE
2 7 POST WAFER CONNECTORS (.156" PIN CENTERS) (J6 AND J8)
4 EDGE BOARD CONNECTORS, 43/86 PINS ON .156" CENTERS (J2-J5)
12 WIRE WRAP POSTS
4 10 PIN, 2.2K, 9 RES, 1.5W RESISTOR PACKS (RP1-RP4)
1 10 PIN, 1K, 9 RES, 1.5W RESISTOR PACK (RP5)
1 10 PIN, 1.1K, 9 RES, 1.5W RESISTOR PACK (RP6)
1 1K RESISTOR, 1/8W, ±5% (R1)
1 2.2K RESISTOR, 1/8W, ±% (R5)
2 220Ω RESISTORS, 1/4W, ±5% (R9, R11)
2 330Ω RESISTORS, 1/4W, ±5% (R10, R12)
2 510Ω RESISTORS, 1/8W, ±5% (R7, R8)

Figure 5. MULTIBUS® System Backplane Example

PHYSICAL CHARACTERISTICS (Con't)

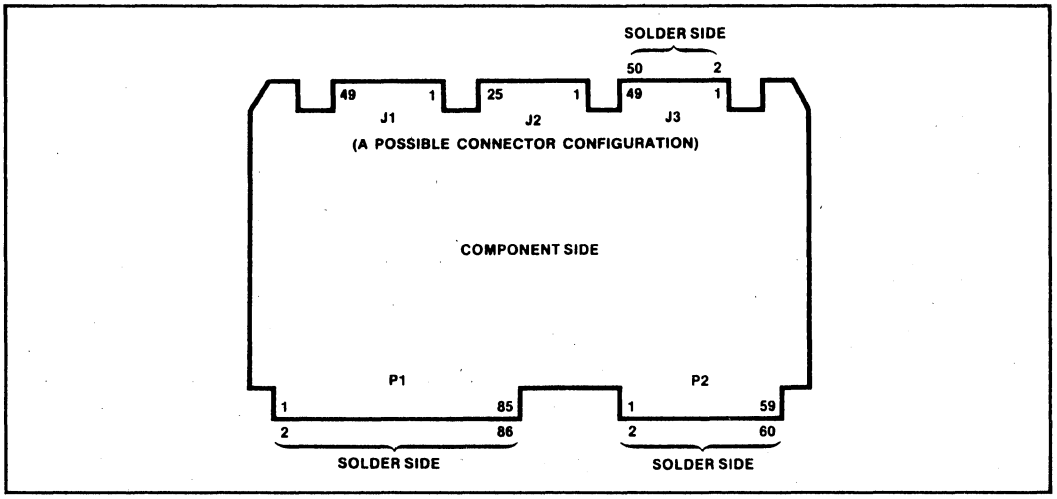


Figure 6. Connector and Pin Numbering

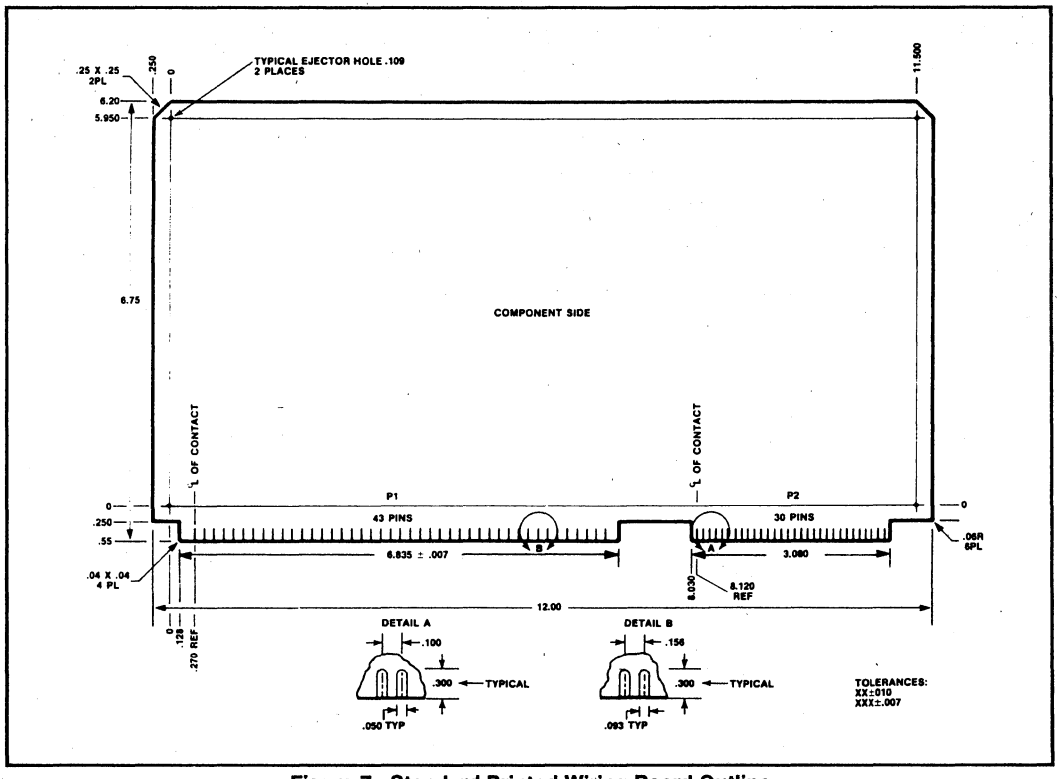


Figure 7. Standard Printed Wiring Board Outline

Backplane Connectors

Table 4. Connector Vendors

Function	# Of Pins	Centers Inches	Connector Type	Vendor	Vendor #	Intel #
Multibus Connector (P1)	43/86	0.156	Soldered[1]	VIKING ELFAB	2KH43/9AMK12 BS1562D43PBB	102247-001
Multibus Connector (P1)	43/86	0.156	Wire wrap[2]	ELFAB ELDAC	BW1562D43PBB 3370860540201	102248-001
				ELFAB EDAC	BW1562A43PBB 337086540202	102273-001[3]
Auxiliary Connector (P2)	30/60	0.1	Soldered[1]	ELFAB EDAC	BS1020A30PBB 345060524802	102238-001
Auxiliary Connector (P2)	30/60	0.1	Wire wrap[2]	TI VIKING	H421121-30 3KH30/9JNK	N/A[3]
				EDAC ELFAB	345060540201 BW1020D30PBB	102241-001

Notes:
1. Connector heights are not guaranteed to conform to Intel packaging equipment.
2. Wirewrap pin lengths are not guaranteed to conform to Intel packaging equipment.
3. With mounting ears with .128 mounting holes.

Environmental Characteristics

Operating Temperature — 0 to 55°C; free moving air across modules and bus

Humidity — 90% maximum (no condensation)

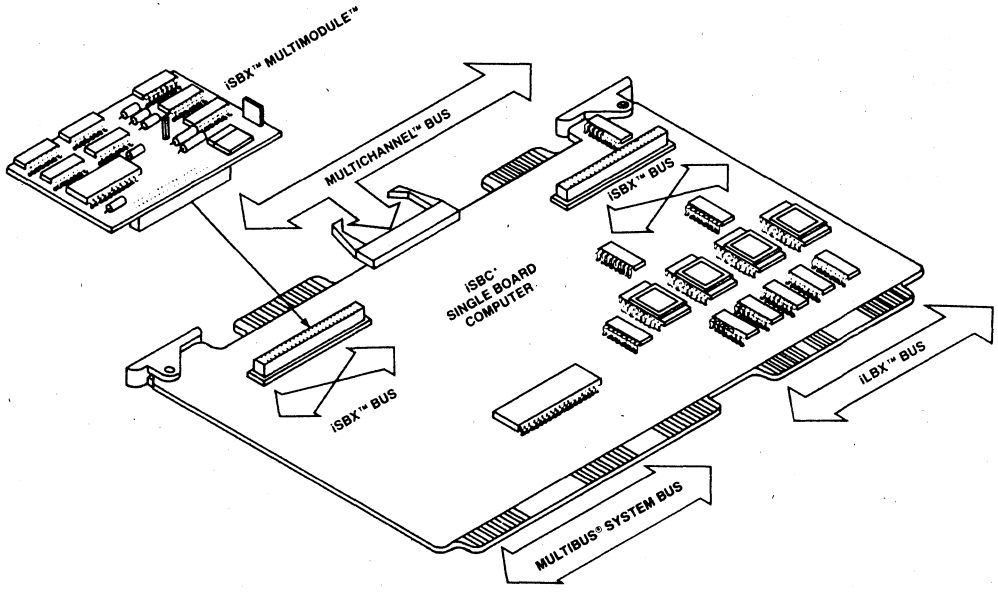
Reference Manuals

210883 — MULTIBUS Handbook

MULTICHANNEL™ I/O BUS

- **High speed 8- or 16-bit block transfers between memory and/or I/O**
- **Transfer rates up to 8 megabytes/sec.**
- **Full speed operation at distances of up to 15 meters.**
- **Supports Supervisor, Controller, or basic Talker/Listener capabilities**
- **Off-loads burst mode I/O activities from host CPU**
- **Up to 16 devices may be interfaced to the bus.**
- **16 megabytes of memory and 16 megabytes of I/O are addressable on each device**

The MULTICHANNEL™ I/O Bus is one of a family of standard bus structures resident within Intel's total system architecture. The MULTICHANNEL bus is a general purpose, high-speed I/O bus capable of significantly increasing system performance by providing a separate data path for DMA I/O activities. By isolating I/O transfers from the system bus, the MULTICHANNEL bus off-loads I/O activity from the host CPU, reduces the probability of bus saturation on the system bus, and reduces contention between I/O and data processing activities on the system bus. The MULTICHANNEL bus can support up to 16 devices at distances up to 15 meters with a maximum burst throughput of 8 megabytes per second. These 16 devices are classified in a manner similar to the IEEE 488 bus concept: Supervisors, Controllers, or Talker and Listeners. As a non-proprietary, standardized I/O bus, the MULTICHANNEL bus is a cost-effective DMA interface ideal for applications such as computer graphics, specialized peripheral control, automatic test equipment, video camera image processing, data acquisition, and high-speed MULTIBUS® system-to-system communication.



FUNCTIONAL DESCRIPTION

Architectural Overview

The MULTICHANNEL bus is the standard high speed I/O interface to MULTIBUS-based systems. Its general purpose design and high performance (8 MB/sec) augment the overall system design by improving I/O interface flexibility and system throughput. The flexibility is realized by using an easy-to-use public standard interface that can support up to sixteen 8-bit or 16-bit devices at up to 15 meters. This structure allows the MULTICHANNEL bus to provide easy I/O system expansion, effective box-to-box communication, and a growth path capable of supporting new generations of high-performance I/O devices. The MULTICHANNEL bus increases system throughput by providing a high-performance data path for efficient movement of large amounts of data.

Structural Features

MULTICHANNEL™ BUS CONFIGURATION

The MULTICHANNEL bus is a multiplexed, asynchronous block transfer, 16-bit I/O bus designed to handle 8-bit and 16-bit transfers between peripherals and single board computers. Its structure (pictured in Figure 2) consists of 16 address/data lines, 6 control lines, 2 interrupt lines, plus parity and reset. These signal lines are implemented as either a 60 conductor flat ribbon cable or a twisted-pair cable spanning a distance of up to 15 meters. A 30/60-pin 3M® connector is recommended for device connection to the MULTICHANNEL bus. The male connectors are installed on each MULTICHANNEL device and the female connectors are mounted on the cable. To insure system integrity, the MULTICHANNEL cable is terminated at both ends.

BUS ELEMENTS

Three device types — the Basic device, the bus Controller device, and the bus Supervisor device — each provide a different level of capability. The Basic Talker/Listener device has lowest capability, responding only to data transfer requests issued by a Supervisor or Controller. The bus Controller device has higher capability than a Basic Talker/Listener on the bus. It can respond to data transfer requests, control data transfers, and can program other MULTICHANNEL devices under direction from a bus Supervisor. Operating at the highest capability is the bus Supervisor device. It provides major control and management of the MULTICHANNEL bus. The bus Supervisor resolves and grants MULTICHANNEL bus priority, monitors bus status, handles interrupts, and controls the reset line, in addition to performing all bus Controller functions.

3M is a registered trademark of 3M Corporation.

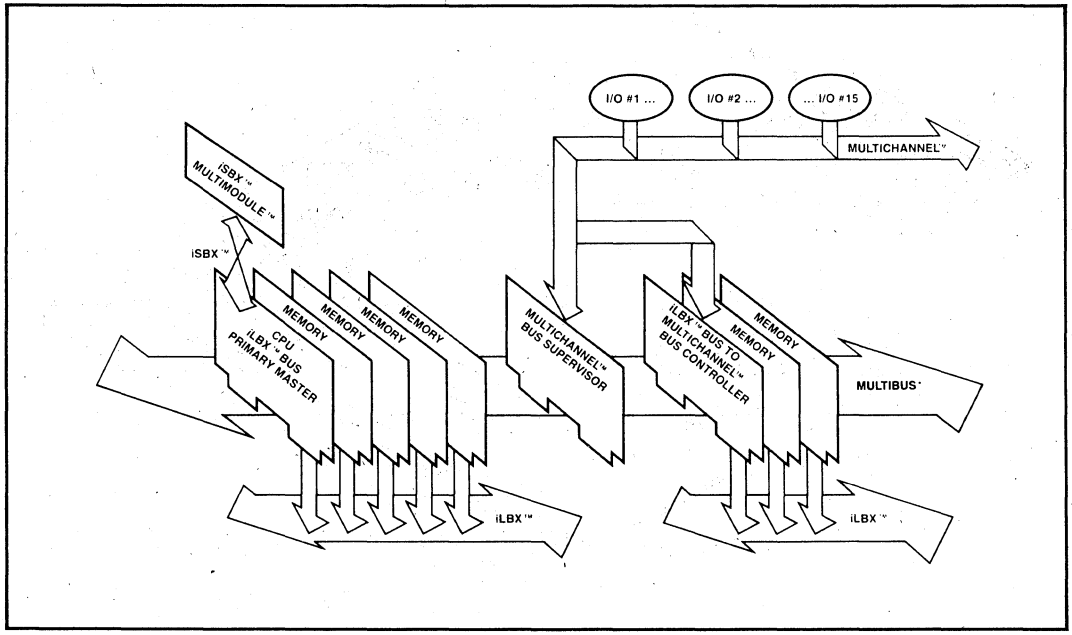


Figure 1. MULTIBUS® System Architecture

MULTICHANNEL bus devices are functionally flexible, creating overlaps between types of bus functions and types of bus devices performing those functions. These devices perform functions in various states of operation: master, slave, talker, listener. When a device is controlling the command/action lines, it is in the master state, and both the bus Supervisor and the bus Controller can operate in this state, although not simultaneously. The slave state indicates a device that can monitor the command/action lines. Only Controllers and Basic Talker/Listeners operate as slaves. All three device types can operate in the talker state or the listener state, but not all at the same time. A Talker is any device selected by the bus master which is writing data to the bus. A Listener is any selected device which is reading data from the bus.

BUS INTERFACE/SIGNAL LINE DESCRIPTIONS

The MULTICHANNEL bus signal lines are grouped into five classes based on the functions they perform: address/data, control, interrupt, parity, and reset. The 16 address/data lines are multiplexed by a control line to act either as 16 unidirectional address lines or 16 bidirectional data lines. When used as address lines, they transmit the device address to all devices attached to the MULTICHANNEL bus. When used as bidirectional data lines, they transmit and receive data to or from MULTICHANNEL devices. The six control lines determine the overall operation of the bus from specifying the type of data transfer to providing the handshake for data transfers between MULTICHANNEL devices. Two interrupt lines are supplied to initiate and terminate data transfers, and to indicate device failures, memory failures, or parity errors. A parity line and a reset line provide support for a parity option and system reset capability whenever required.

BUS PIN ASSIGNMENTS

For proper MULTICHANNEL implementation, a 60 conductor (twisted pair or flat) cable using a 30/60 pin 3M connector, is used for device connection to the bus. Figure 3 is an outline drawing of the iSBC® MULTICHANNEL connector which also shows the pin numbering. The MULTICHANNEL bus connector signal pin assignments are listed in Table 1. Cable termination is implemented at both cable ends to insure proper system integrity over a 15-meter cable. Figure 4 is a schematic of the cable termination circuits. A cable termination module could be created that would then be connected to the cable end via a 30/60 pin connector.

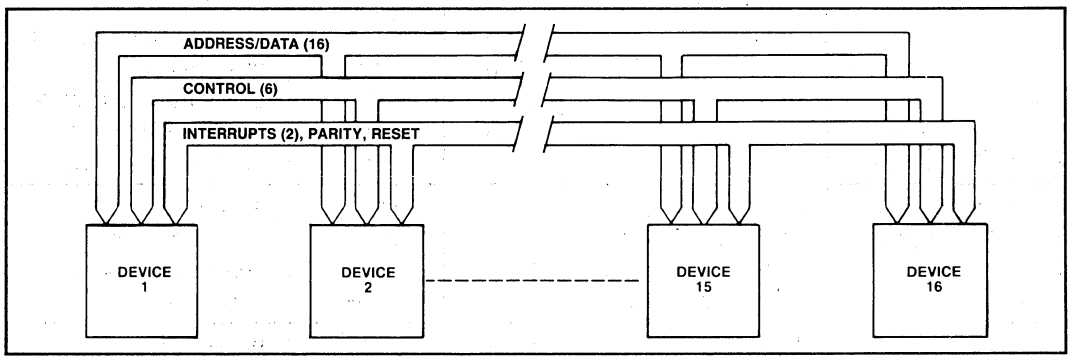


Figure 2. Block Diagram of MULTICHANNEL™ Bus Structure

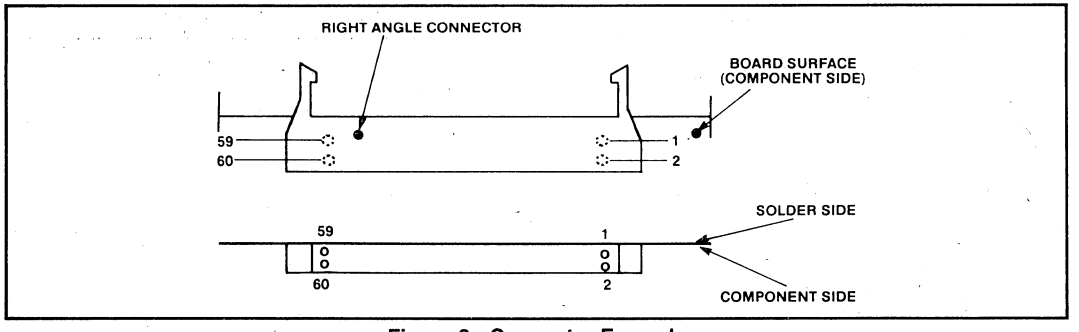


Figure 3. Connector Example

Table 1. MULTICHANNEL™ Bus Pin Assignments

Lower Row			Upper Row		
Pin	**Mnemonic**	**Signal Name**	**Pin**	**Mnemonic**	**Signal Name**
1	GND	GROUND	2	AD0/	ADDRESS DATA LINE 0
3	GND	GROUND	4	AD1/	ADDRESS DATA LINE 1
5	GND	GROUND	6	AD2/	ADDRESS DATA LINE 2
7	GND	GROUND	8	AD3/	ADDRESS DATA LINE 3
9	GND	GROUND	10	AD4/	ADDRESS DATA LINE 4
11	GND	GROUND	12	AD5/	ADDRESS DATA LINE 5
13	GND	GROUND	14	AD6/	ADDRESS DATA LINE 6
15	GND	GROUND	16	AD7/	ADDRESS DATA LINE 7
17	GND	GROUND	18	AD8/	ADDRESS DATA LINE 8
19	GND	GROUND	20	AD9/	ADDRESS DATA LINE 9
21	GND	GROUND	22	ADA/	ADDRESS DATA LINE 10
23	GND	GROUND	24	ADB/	ADDRESS DATA LINE 11
25	GND	GROUND	26	ADC/	ADDRESS DATA LINE 12
27	GND	GROUND	28	ADD/	ADDRESS DATA LINE 13
29	GND	GROUND	30	ADE/	ADDRESS DATA LINE 14
31	GND	GROUND	32	ADF/	ADDRESS DATA LINE 15
33	GND	GROUND	34	RESET/	RESET
35	GND	GROUND	36	AACC	ADDRESS MODE ACCEPT
37	GND	GROUND	38	SRQ/	SERVICE REQUEST
39	GND	GROUND	40	STO/	SUPERVISOR TAKE OVER
41	GND	GROUND	42	DACC/	DATA MODE ACCEPT
43	GND	GROUND	44	SA/	SUPERVISOR ACTIVE
45	PB*/	PARITY BIT (INV.)	46	PB/	PARITY BIT
47	R/W/	READ NOT WRITE (INV.)	48	R/W	READ NOT WRITE
49	A/D/	ADDRESS NOT DATA (INV.)	50	A/D	ADDRESS NOT DATA
51	DRDY*/	DATA READY (INV.)	52	DRDY/	DATA READY
53	RES	RESERVED	54	RES	RESERVED
55	RES	RESERVED	56	RES	RESERVED
57	RES	RESERVED	58	RES	RESERVED
59	RES	RESERVED	60	RES	RESERVED

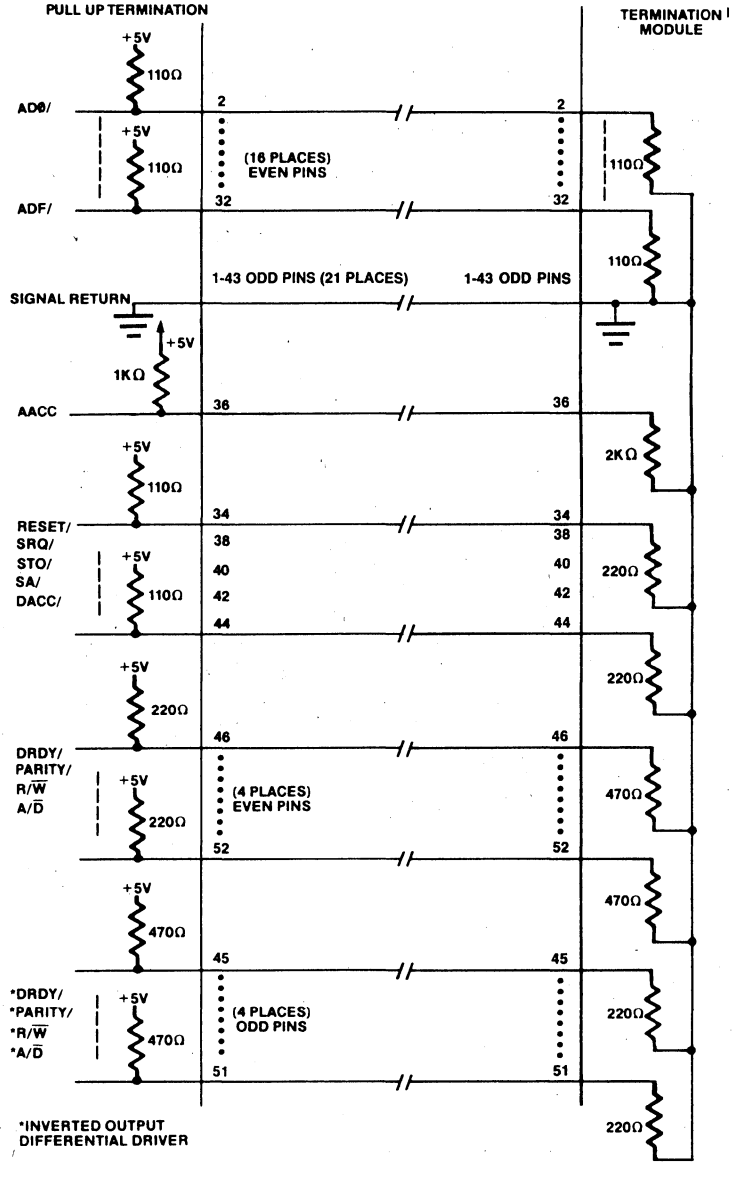


Figure 4. Bus Termination Schematic

Bus Operation Protocol

DATA TRANSFER OPERATION

There are three modes of communication in the operation protocol: address mode, data mode, and control transfer mode. Using these transfer modes, each MULTICHANNEL device provides handshaking capability for totally asynchronous block data transfers. Address mode is the time when the address/data control line is high. Information placed on the address/data lines of the MULTICHANNEL bus as two successive 16-bit words is interpreted to select or deselect a device on the bus and address the specific resource on the device. Typically, these address mode transfers are only 2 word sequences. Figure 5 is a timing diagram of the handshake routine in address mode. The data mode is the time when the address/data control line is low. Valid data is placed on the address/data lines of the bus and can occur only after an address mode has been performed. Transfers during data mode are usually large quantities of either 8- or 16-bit data, and are passed to or from the addressed device until it is deselected. Figure 6 is a timing diagram of a data transfer sequence.

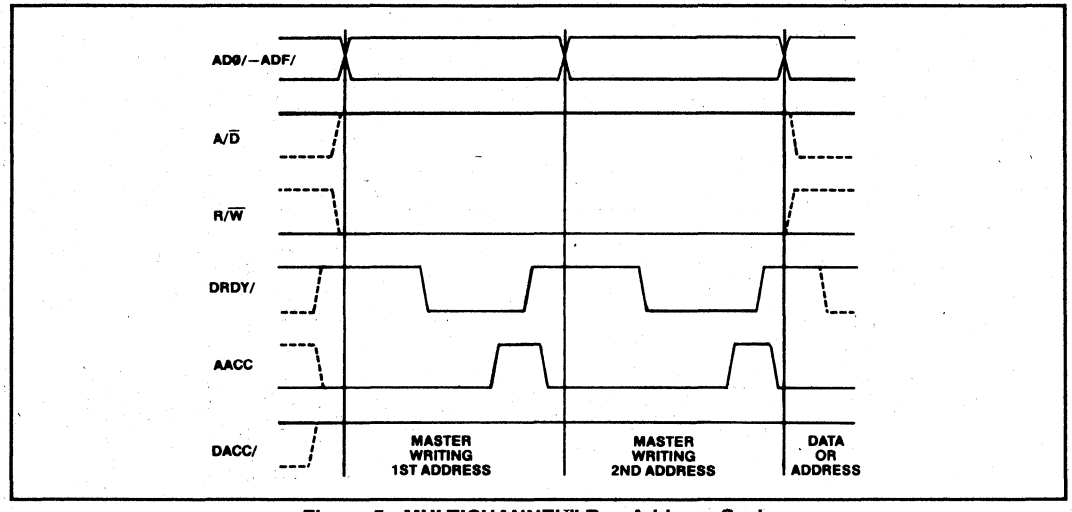


Figure 5. MULTICHANNEL™ Bus Address Cycle

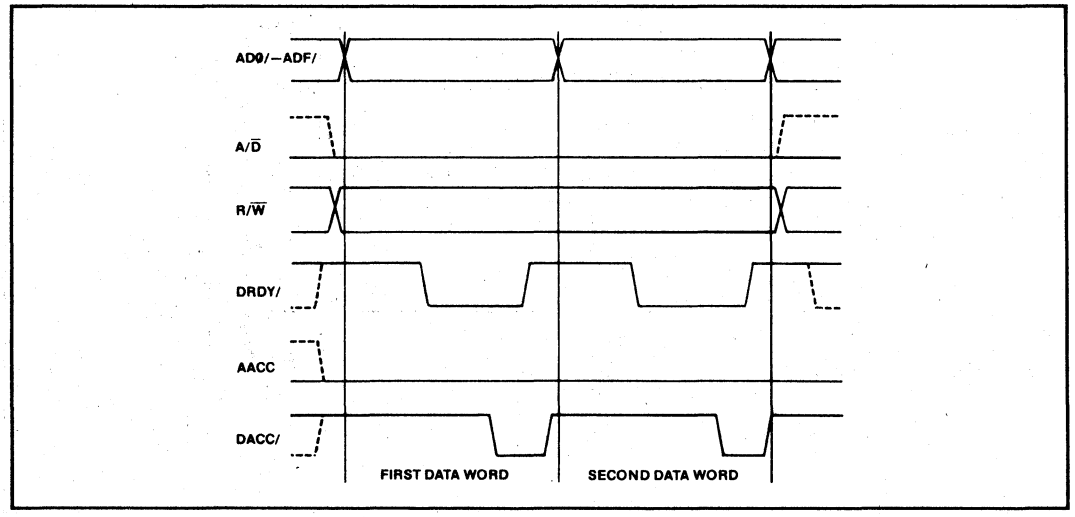


Figure 6. MULTICHANNEL ™ Bus Data Transfer Sequence

Control transfer mode is the time when the bus Supervisor selects the bus Controller and programs its registers with required information. Once programmed, a bus Controller may select a device and originate a data transfer operation.

The operational sequences of these transfer modes are similar in handling read and write operations to and from the 16 megabytes of memory and the 16 megabytes of registers addressable on each MULTICHANNEL device.

A typical transfer sequence begins when the master sends a two-word address sequence to select a MULTICHANNEL device and specify address, direction and resource (memory vs. I/O) of the data transfer. Following device selection, the Talker proceeds to send the data as a continuous 8 or 16-bit data word stream until the block data move is complete. The master terminates the transfer by issuing another two-word address sequence for device deselection.

The transfer sequence described is identical for both memory and register type transfers. The master controls similar read and write operations between devices, and the address select and deselect sequences use the same address format. Figure 7 contains the MULTICHANNEL bus address format.

DEVICE REGISTER DEFINITION

Of the 16 megabytes of register space per device, the first 16 registers are pre-defined to provide a standard register area common to all devices. The remaining registers are user definable. Table 2 lists the 16 defined registers along with their function. The use of this register concept allows for standard interface between all MULTICHANNEL devices. Please refer to the MULTICHANNEL Bus Specification for more detailed information.

Table 2. MULTICHANNEL™ Device Register Definitions

Register Number	Definition	Mode
0	STO/ Flag/Status	Read Only
1	SRQ/ Flag/Status	Read Only
2	SRQ/ Mask	Write Only
3	Device Command	Write Only
4	Device Parameter	Write Only
5	Data Address 1	Read or Write
6	Data Address 2	Read or Write
7	Block Length 1	Read or Write
8	Block Length 2	Read or Write
9	Error Address 1	Read Only
10	Error Address 2	Read Only
11	Address Extension	Write Only
12-15	Reserved	
16-16 Mbyte	User Defined	Read or Write

BUS INTERRUPT HANDLING

The MULTICHANNEL bus Supervisor, being responsible for bus access and control, monitors the two bus interrupt lines. The Supervisor Take-Over interrupt (STO) is used to inform the bus Supervisor that a device wants to return control of the bus to the Supervisor or that an error has occurred. The Service Request Interrupt (SRQ) is used by devices which do not have control of the bus, but require service from the bus Supervisor. To locate a device transmitting a bus interrupt, the bus Supervisor

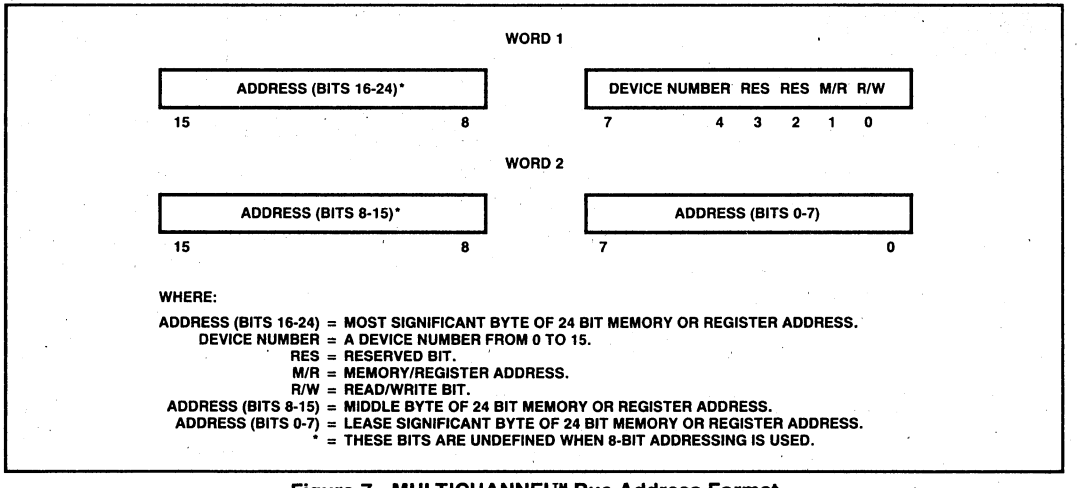


Figure 7. MULTICHANNEL™ Bus Address Format

polls each device attached to the bus by reading the appropriate register of each device and testing for a non-zero value. In current implementations, the Supervisor polls each device only once. If the interrupt is not removed an error occurs.

PARITY AND RESET

Parity operation on the MULTICHANNEL bus is provided, but is not required. The bus Supervisor selects between parity mode and non-parity mode depending upon system requirements. If parity mode is selected all Talkers must generate odd parity. All active Listeners monitor the parity line and generate an STO interrupt signal if there is a parity error.

A reset function is also supported by the MULTICHANNEL bus, and is controlled by the bus Supervisor to bring the bus to a known state. It is used to reset all devices after power-up, and when required to gain control of the bus.

SPECIFICATIONS

Word Size

Data — 8, 16-bit

Memory Addressing

24-bits — 16 megabyte – direct access – automatic incrementing

Register Addressing

24-bits — 16 megabyte – direct access

Electrical Characteristics

DC SPECIFICATIONS

Maximum Bus Length

15 meters (50 feet)

Bus Devices Supported

16 total devices — (Supervisor, Controller, and Talker/Listener)

Bus Bandwidth

8 megabytes/sec. — 16-bit

4 megabytes/sec. — 8-bit

Table 3. DC Specifications

Signal Name	Driver Type	Termination (see Note)	Min. Driver Requirements			Max. Receiver Requirements		
			High	Low	Load Cap	High	Low	Load Cap
AD15-0/	TRI-STATE	110 Ohms	– 5 ma	48 ma	300 pf	0.2 ma	0.8 ma	15 pf
SA/	OPEN COLL	110/220 Ohms	N.A.	48 ma	300 pf	0.4 ma	0.6 ma	15 pf
RESET/	OPEN COLL	110/220 Ohms	N.A.	48 ma	300 pf	0.4 ma	0.6 ma	15 pf
AACC	OPEN COLL	1K/2K Ohms	N.A.	48 ma	300 pf	0.4 ma	0.6 ma	15 pf
DACC/	OPEN COLL	110/220 Ohms	N.A.	48 ma	300 pf	0.4 ma	0.6 ma	15 pf
SRQ/	OPEN COLL	110/220 Ohms	N.A.	48 ma	300 pf	0.4 ma	0.6 ma	15 pf
STO/	OPEN COLL	110/220 Ohms	N.A.	48 ma	300 pf	0.4 ma	0.6 ma	15 pf
R/W	DIF, NON-INV	220/470 Ohms	– 20 ma	40 ma	300 pf	0.5 ma	0.5 ma	15 pf
R/W/	DIF, INV	470/220 Ohms	– 20 ma	40 ma	300 pf	0.5 ma	0.5 ma	15 pf
A/D	DIF, NON-INV	220/470 Ohms	– 20 ma	40 ma	300 pf	0.5 ma	0.5 ma	15 pf
A/D/	DIF, INV	470/220 Ohms	– 20 ma	40 ma	300 pf	0.5 ma	0.5 ma	15 pf
PB/	DIF, NON-INV	220/470 Ohms	– 20 ma	40 ma	300 pf	0.5 ma	0.5 ma	15 pf
PB*/	DIF, INV	470/220 Ohms	– 20 ma	40 ma	300 pf	0.5 ma	0.5 ma	15 pf
DRDY/	DIF, NON-INV	220/470 Ohms	– 20 ma	40 ma	300 pf	0.5 ma	0.5 ma	15 pf
DRDY*/	DIF, INV	470/220 Ohms	– 20 ma	40 ma	300 pf	0.5 ma	0.5 ma	15 pf

NOTE: Termination provided only at the physically ends of the interconnect cable. Where the positive termination (pull-up) resistance is different from the negative termination (pull-down) resistance, the positive termination resistance is listed first.

Cables and Connectors

Table 3. Cable and Receptacle Vendors

MULTICHANNEL™ Bus Compatible Cable			
Vendor	**Ribbon Type**	**Vendor No.**	**Conductor**
Belden	Plain Flat	9L28060	60
Belden	Twisted-Pair	9V28060	60
Belden	Insulated Flat	9L28260	60
Spectrastrip	Plain Flat	455-240-60	60
Spectrastrip	Twisted-Pair	455-248-60	60
Spectrastrip	Insulated Flat	151-2830-060	60
MULTICHANNEL™ Bus Compatible Receptacles			
Vendor	**Type**	**Vendor No.**	**Pins**
Berg	Male	65823-103	60
Berg	Female	65949-960	60
3M	Male	3372-1302	60
3M	Female	3334-6000	60

PHYSICAL PROPERTIES

Conductors — 28 AWG, 7/36 strand, tinned copper

Conductor Insulation — 0.010 inch wall, nominal

Conductor Spacing — Twisted pair – 0.10 inch, nominal; Flat – 0.050 inch, ±10%

Cable Thickness — Flat – 0.042 inch, nominal

Temperature Rating — 80°C

ELECTRICAL PROPERTIES

Impedance (nominal) — 105 ohms ±10%

Propagation Velocity (nominal) — 1.7 ns/ft

Capacitance (nominal) — 22 pf/ft

INSULATION REQUIREMENTS

Voltage Rating (minimum) — 100 Vdc

Insulation Resistance (minimum) — 1×10^{10} ohms

Environmental Characteristics

Temperature — 0 - 55°C

Humidity — 90% max. relative (no condensation)

Reference Manuals

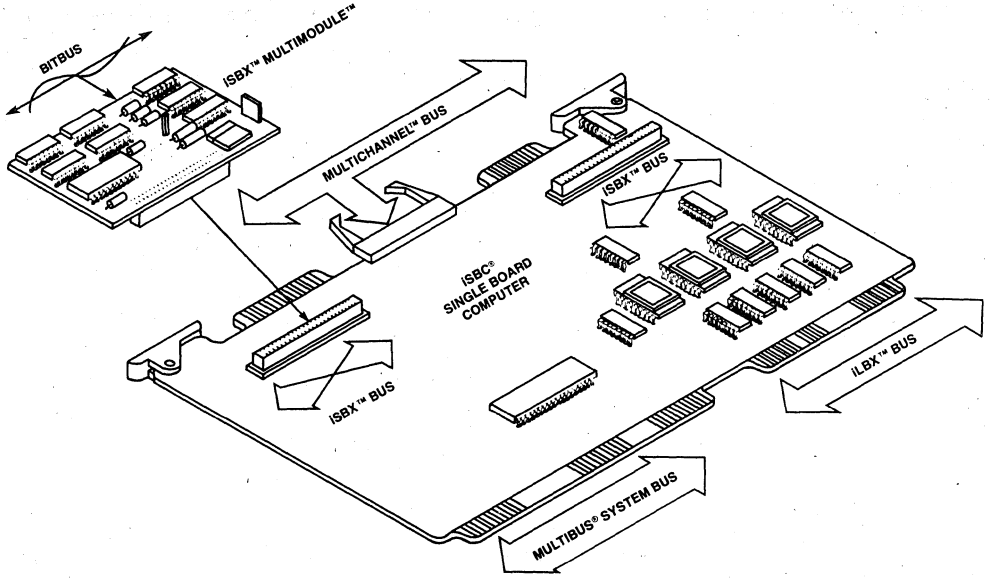
210883 — MULTIBUS Architecture Handbook.

PRELIMINARY

iLBX™ EXECUTION BUS

- **High bus bandwidth**
  - **— 9.5 Mbytes/sec. for 8-bit transfers**
  - **— 19 Mbytes/sec. for 16-bit transfers**
- **16 Mbyte addressing range**
- **8 and 16-bit data transfers**
- **Supports up to 5 iLBX™ compatible devices per bus**
- **Primary and secondary master bus exchange capabilities**
- **Standard 60-pin MULTIBUS® P2 connector**

The iLBX™ Execution Bus is one of a family of standard bus structures resident within Intel's total system architecture. The Local Bus Extension (iLBX) Bus is a dedicated execution bus capable of significantly increasing system performance by extending the processor board's on-board local bus to off-board resources. This extension provides for arbitration-free, direct access to high-performance memory. Acting as a "virtual" iSBC®, up to 16 megabytes of processor addressable memory can be accessed over the iLBX bus and appear as though it were resident on the processor board. The iLBX Bus preserves advantages in performance and architecture of on-board memory, while allowing memory configurations larger than possible on a single board computer. High throughput and independence from MULTIBUS® activities make the iLBX bus an ideal solution for "working store" type program memory and data processing applications requiring large amounts of high performance memory. Such applications include graphics systems, robotics, process control, office systems, and CAD/CAM.



December, 1982
Order Number: 210893-002

FUNCTIONAL DESCRIPTION

Architectural Overview

The iLBX bus is an architectural solution for supporting large amounts of high performance memory. It is the first structure that allows the CPU board selection to be decoupled from the on-board memory requirement, and still maximizes the processor's performance potential. It eliminates the processor's need to access its off-board memory resources solely over the MULTIBUS system bus. Architectural consistency with the single board computer approach including iLBX memory can be maintained by dual port access of memory resources between the iLBX bus and the MULTIBUS system bus. This allows for global access by other processors and I/O devices while still providing high speed local CPU operations. This sub-system created by the iLBX bus of a single board computer and a maximum of 4 memory cards can be perceived architecturally as a "virtual single board computer". The implementation of iLBX bus "virtual modules" makes it possible to create functional modules with a new level of flexibility and performance in implementing a wide range of memory capabilities. With future needs in mind, the iLBX bus has the capability of accessing a full 16 megabytes of memory.

Structural Features

The iLBX bus uses a non-multiplexed 16-bit configuration capable of 8 and 16-bit transfers. Used in conjunction with the MULTIBUS interface, the iLBX bus resides on the MULTIBUS form factor P2 connector and supercedes the MULTIBUS interface definitions for the P2 signals. The iLBX bus uses the standard 60-pin MULTIBUS P2 connector and occupies 56 of the P2 connector pins with I6 data lines, 24 address lines plus control, command access, and parity signals. The four MULTIBUS address extension lines on the MULTIBUS/iLBX P2 connector retain the standard MULTIBUS interface definition.

Bus Elements

The iLBX bus supports three distinct device categories: 1) Primary Master, 2) Secondary Master, 3) Slave. These three device types may be combined to create several iLBX local busses ranging (in size) from a minimum of two to a maximum of five devices per iLBX bus. There is only one Primary Master in any given implementation of iLBX bus, and its presence is required along with the attachment of at least one Slave device. To provide alternate access over an iLBX bus, one optional Sec-

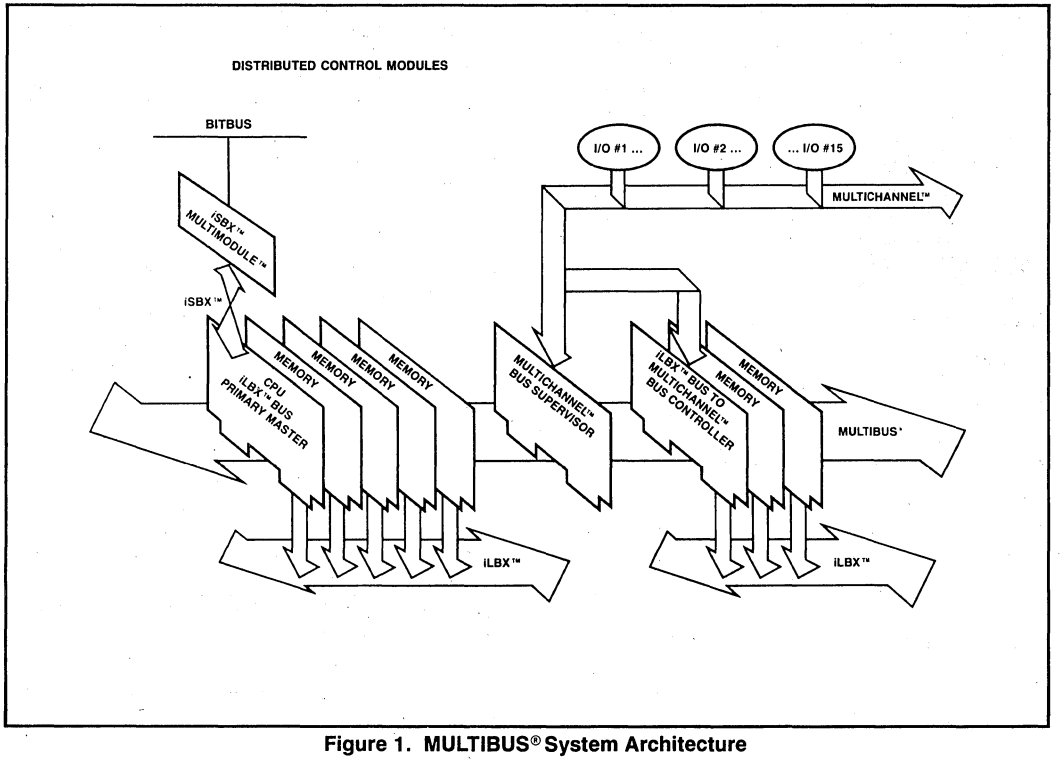


Figure 1. MULTIBUS® System Architecture

ondary Master may be incorporated to create a "two-master" local bus subsystem. By limiting the iLBX bus to two masters (a Primary and a Secondary), bus arbitration is reduced to a simple request and acknowledge process, with privileged use of the bus maintained by the Primary Master, and limited access granted to the Secondary Master when needed.

The Primary Master executes the role of iLBX bus "supervisor" by controlling the general operation of the bus and managing Secondary Master accesses to the Slave memory resources.

The Secondary Master Device is an option providing alternate access to the Slave resources on the iLBX bus. Secondary master devices are typically DMA driven. This feature is provided for implementation flexibility when occasional DMA transfers in and out of iLBX memory resources can optimize the overall system performance. The Secondary Master essentially duplicates the Primary Master's data transfer capability, but must rely on the Primary Master to grant access. Once access is granted, the Secondary Master controls the bus, and drives all signal lines until the operation is complete and control is passed back to the Primary Master.

The Slave devices contain the memory resources used by the Primary Master and the optional Secondary Master. Each iLBX implementation can contain a maximum of four Slave devices. Using 64K RAM technology on four slave devices with ECC can provide for over 2 megabytes of "on-board" high performance memory. With 256K RAM chips, each iLBX bus could contain slave devices with memory totalling 8 megabytes. As memory technology increases, the iLBX bus is designed to incorporate it in rapid fashion because it is capable of directly accessing a full 16 megabytes of memory on its high-performance Slave devices.

Bus Interface/Signal Line Descriptions

The iLBX bus interface is divided into four functional classes of signal lines: address and data lines, control lines, command lines, and bus access lines. The 40 address and data lines defined by the iLBX Bus Specification consist of 16 data lines and 24 address lines.

There are 16 bi-directional data lines exclusively used to handle 8-bit and 16-bit data transfers between the active bus master and the selected slave device. The iLBX bus uses these data lines for all data transfers, and are driven by tri-state drivers.

The 24 address lines on the iLBX bus provide the ability to directly address 16 megabytes of memory. These single-direction address lines are exclusively driven by the active bus master. The iLBX bus master uses them to select a specific slave device. Three control lines specify the type of data transfer between master and slave devices, while the three command lines initiate, control, and terminate the transfer. There are also three bus access lines used to transfer bus control between master devices.

Bus Pin Assignments

The iLBX bus uses the standard 60-pin MULTIBUS P2 connector. The physical location of each pin assignment and its corresponding function is listed in Table 1.The four MULTIBUS address extension lines (pins 55-58 on the P2 connector) retain the standard MULTIBUS interface functions.

Bus Operation Protocol

The operation protocol for the iLBX bus is a straightforward set of procedures consisting of three basic operations: bus control access, write data to memory, read data from memory. These operations use asynchronous protocol with positive acknowledgment.

Bus Access

The iLBX bus is shared by at most two masters; one Primary Master and one optional Secondary Master, each providing an alternate access path to iLBX bus memory resources. The mechanism for obtaining bus access is a simple request and acknowledge process communicated between masters. Each master is a bus controller of similar capabilities, responsible for data transfer operations between devices, but the Primary Master has the added responsibility of controlling iLBX bus accesses.

The Primary Master has default control of the iLBX bus. If the Secondary Master needs access to the bus, it must initiate a request and wait for acknowledgment from the Primary Master. The choice of when to surrender control of the bus rests with the Primary Master, but if no data transfer is in progress, the Primary Master normally relinquishes control immediately to the Secondary Master.

Data Transfer Operation

The iLBX bus supports two types of data transfer operations: write data to memory and read data from memory. These data transfer operations facilitate the passing of information between the active bus master and the selected slave device. The operation of these two transfer types is very similar; the only differences being the direction of the data transfer and the device driving the data lines.

For either type of data transfer, the active bus master first initiates the transfer operation by placing the memory address on the address lines (AB23-AB0) and a con-

trol configuration on the control lines to select the slave device. Once the slave device is selected, the type of data transfer becomes the key factor. With the write operation, the active master maintains control of the data lines and provides valid data within the specified time. Upon accepting a data element, the slave sends a receipt acknowledgment signal to the master which completes the data transfer operation.

With the read operation, the slave device drives the data lines and places valid data on the data lines before sampling by the active master. The slave acknowledges the master to signal the end of the data transfer, and the master completes the operation.

The iLBX Bus Specification includes provisions for both optimized and non-optimized data transfers. Optimized

Table 1. iLBX™ Bus Pin Assignments, P2 Edge Connector

Component Side			Solder Side		
16-Bit Pin	**Mnemonic**	**Signal Name**	**16-Bit Pin**	**Mnemonic**	**Signal Name**
1	DB0	DATA LINE 0	2	DB1	DATA LINE 1
3	DB2	DATA LINE 2	4	DB3	DATA LINE 3
5	DB4	DATA LINE 4	6	DB5	DATA LINE 5
7	DB6	DATA LINE 6	8	DB7	DATA LINE 7
9	GND	GROUND	10	DB8	DATA LINE 8
11	DB9	DATA LINE 9	12	DB10	DATA LINE 10
13	DB11	DATA LINE 11	14	DB12	DATA LINE 12
15	DB13	DATA LINE 13	16	DB14	DATA LINE 14
17	DB15	DATA LINE 15	18	GND	GROUND
19	AB0	ADDRESS LINE 0	20	AB1	ADDRESS LINE 1
21	AB2	ADDRESS LINE 2	22	AB3	ADDRESS LINE 3
23	AB4	ADDRESS LINE 4	24	AB5	ADDRESS LINE 5
25	AB6	ADDRESS LINE 6	26	AB7	ADDRESS LINE 7
27	GND	GROUND	28	AB8	ADDRESS LINE 8
29	AB9	ADDRESS LINE 9	30	AB10	ADDRESS LINE 10
31	AB11	ADDRESS LINE 11	32	AB12	ADDRESS LINE 12
33	AB13	ADDRESS LINE 13	34	AB14	ADDRESS LINE 14
35	AB15	ADDRESS LINE 15	36	GND	GROUND
37	AB16	ADDRESS LINE 16	38	AB17	ADDRESS LINE 17
39	AB18	ADDRESS LINE 18	40	AB19	ADDRESS LINE 19
41	AB20	ADDRESS LINE 20	42	AB21	ADDRESS LINE 21
43	AB22	ADDRESS LINE 22	44	AB23	ADDRESS LINE 23
45	GND	GROUND	46	ACK*	SLAVE ACKNOWLEDGE
47	BHEN	BYTE HIGH ENABLE	48	R/W	READ NOT WRITE
49	ASTB*	ADDRESS STROBE	50	DSTB*	DATA STROBE
51	SMRQ*	SECONDARY MASTER REQUEST	52	SMACK*	SECONDARY MASTER ACKNOWLEDGE
53	LOCK*	ACCESS LOCK	54	GND	GROUND
55	ADR22*	MULTIBUS® ADDRESS EXTENSION LINE 22	56	ADR23*	MULTIBUS® ADDRESS EXTENSION LINE 23
57	ADR20*	MULTIBUS® ADDRESS EXTENSION LINE 20	58	ADR21*	MULTIBUS® ADDRESS EXTENSION LINE 21
59	RES	RESERVED	60	TPAR*	TRANSFER PARITY

operation uses pipelining and signal overlapping techniques to manage the data transfer timing relationships between the active bus master and the selected slave. The use of signal overlapping requires that every device attached to the iLBX bus provide a means for varying the timing of the slave request and acknowledge signals. The non-optimized operation uses fixed signal sequences, instead of signal overlapping, to assure a valid data transfer, and a device does not need a variable request or acknowledge to read data-valid timing on the iLBX bus. Please refer to the iLBX Bus Specification for detailed descriptions of these transfer operations.

Mechanical Implementation

Because the iLBX bus uses the P2 connector of the MULTIBUS form factor, the iLBX bus "shares" a MULTIBUS chassis with the MULTIBUS backplane system bus in the system design. The iLBX mechanical specifications are synonymous with the MULTIBUS specifications for board-to-board spacing, board thickness, component lead length, and component height above the board. The iLBX bus interconnection can use either flexible ribbon cable or a rigid backplane. The iLBX bus interconnect maximum length is limited to 10 cm (approximately 4 inches); that is sufficient to span 5 card slots across two connected chassis. Figure 2 shows an iLBX bus cable assembly.

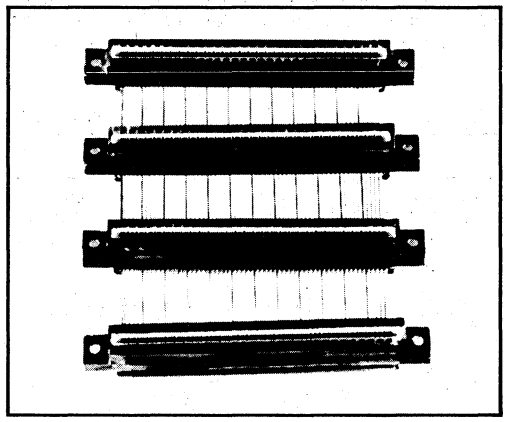


Figure 2. Typical iLBX™ Bus Interface Cable Assembly

SPECIFICATIONS

Word Size

Data — 8 and 16-bit

Memory Addressing

24-bits – 16 megabyte – direct access

Bus Bandwidth

9.5 megabytes/sec — 8-bit

19 megabytes/sec — 16-bit

Electrical Characteristics

DC SPECIFICATIONS

Table 2. DC Specifications

Signal Name	Driver Type	Termination (to +5 Vdc) At Master	Min. Driver Requirements			Max. Receiver Requirements		
			High	Low	Load Cap.	High	Low	Load Cap.
DB15-0	TRI-STATE	10K Ohms	0.4 ma	9 ma	75 pf	0.15 ma	2 ma	18 pf
TPAR*	TRI-STATE	10K Ohms	0.4 ma	9 ma	75 pf	0.15 ma	2 ma	18 pf
AB23-0	TRI-STATE	None	0.4 ma	20 ma	120 pf	0.10 ma	5 ma	30 pf
R/W	TRI-STATE	None	0.2 ma	8 ma	75 pf	0.05 ma	2 ma	18 pf
BHEN	TRI-STATE	None	0.2 ma	8 ma	75 pf	0.05 ma	2 ma	18 pf
LOCK*	TRI-STATE	None	0.2 ma	8 ma	75 pf	0.05 ma	2 ma	18 pf
SMRQ*	TTL	10K Ohms	0.05 ma	8 ma	20 pf	0.05 ma	2 ma	18 pf
SMACK*	TTL	None	0.05 ma	2 ma	20 pf	0.05 ma	2 ma	18 pf
†ASTB*	TRI-STATE	10K Ohms	0.2 ma	9 ma	75 pf	0.05 ma	2 ma	18 pf
†DSTB*	TRI-STATE	10K Ohms	0.2 ma	9 ma	75 pf	0.05 ma	2 ma	18 pf
ACK*	OPEN COLL.	330 Ohms	N.A.	20 ma	45 pf	0.05 ma	2 ma	18 pf

† At slave, series RC termination to GND (100 ohm, 10 pf)

BUS TIMING

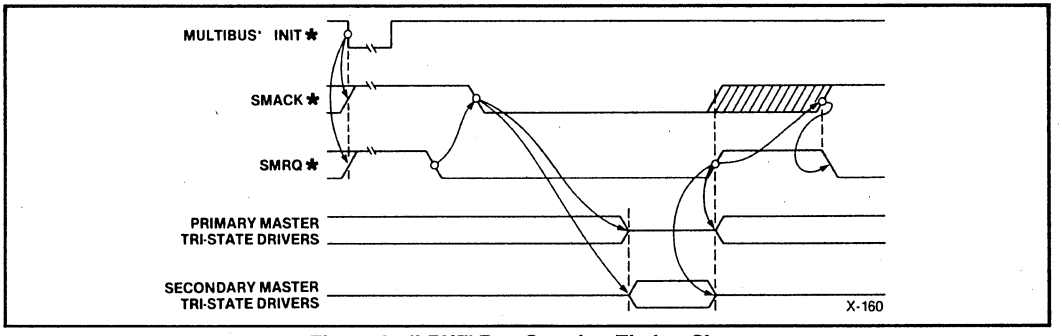


Figure 3. iLBX™ Bus Granting Timing Chart

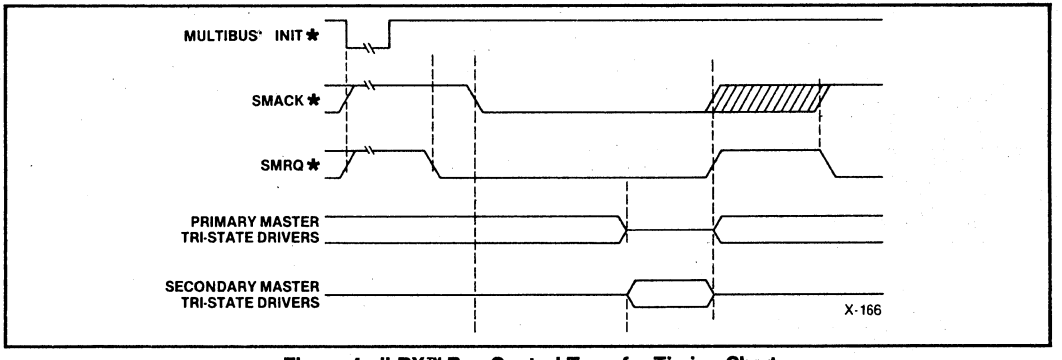


Figure 4. iLBX™ Bus Control Transfer Timing Chart

16-Bit Transfer Timing —

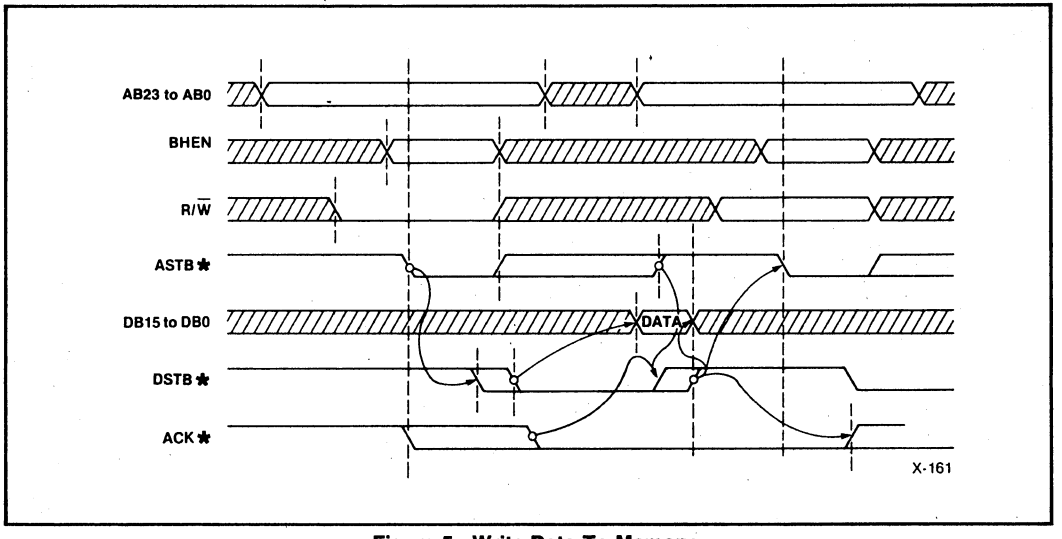


Figure 5. Write Data-To-Memory

BUS TIMING

16-Bit Transfer Timing (Con't.) —

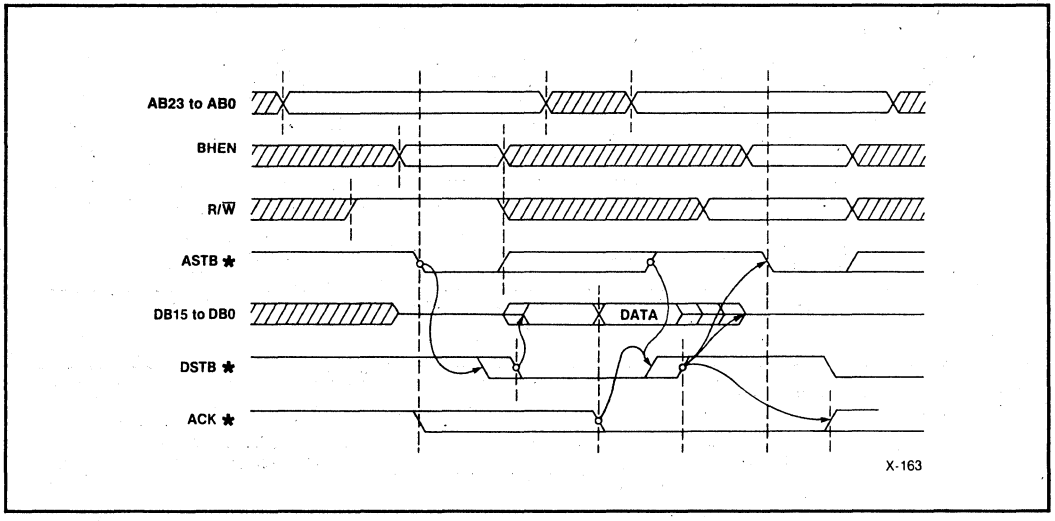


Figure 6. Read Data-From-Memory

Physical Characteristics

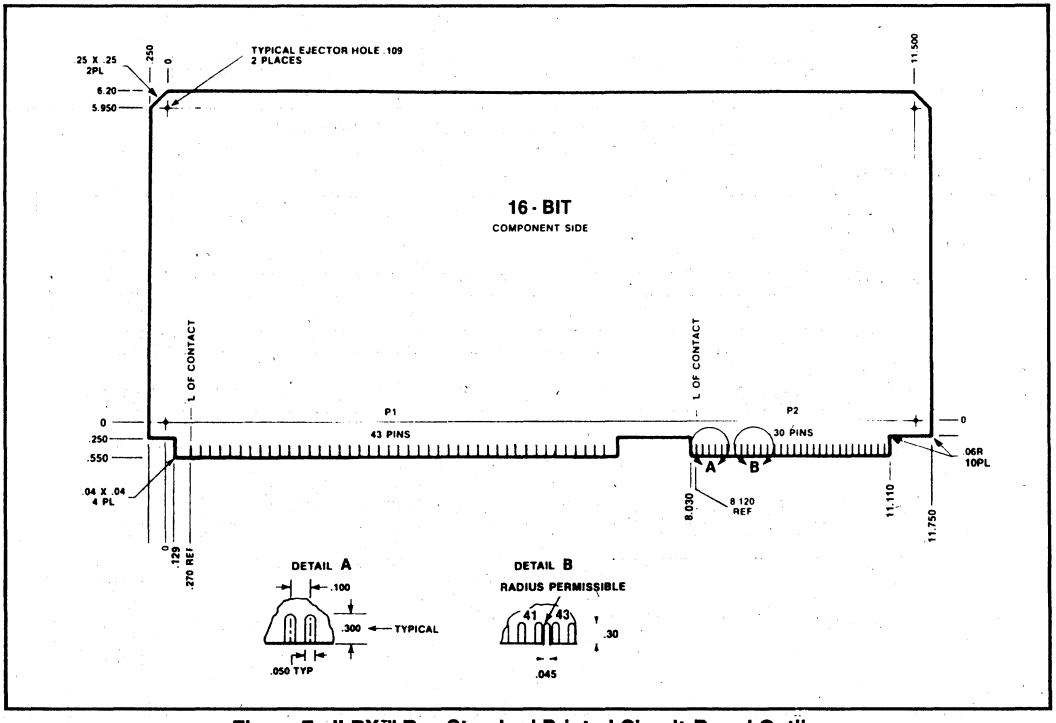


Figure 7. iLBX™ Bus Standard Printed Circuit Board Outline

Cables and Connectors

Table 3. Cable and Receptacle Vendors

iLBX™ Bus Compatible Cable		
Vendor	**Vendor Part No.**	**Conductors**
T&B Ansley	171-60	60
T&B Ansley	173-60	60
3M	3365/60	60
3M	3306/60	60
Berg	76164-060	60
Belden	9L28060	60
Spectrastrip	455-240-60	60
iLBX™ Bus Compatible Receptacles		
Vendor	**Vendor Part No.**	**Pins**
Kelam	RF30-2803-5	60
T&B Ansley	A3020 (609-6025 modified)	60

Environmental Characteristics

OPERATING

Temperature — 0 to 55°C

Relative Humidity — 0 to 85 percent; non-condensing

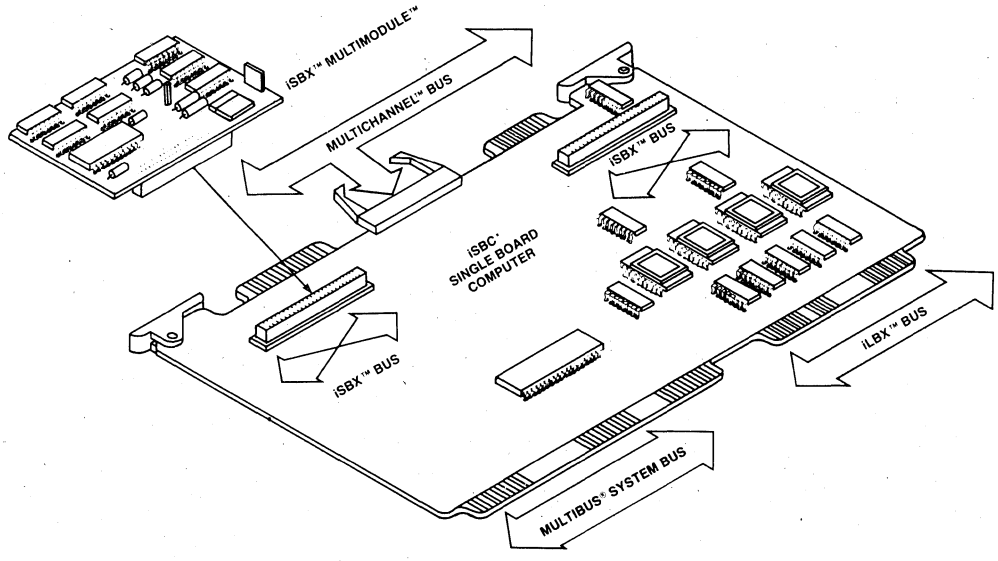
Reference Manuals

210883 — MULTIBUS Handbook

iSBX™ I/O EXPANSION BUS

- **IEEE P959 industry standard I/O expansion bus**
- **Provides on-board expansion of system resources**
- **Small iSBX™ MULTIMODULE™ boards plug directly into iSBC® boards**
- **Supports compatible 8- and 16-bit data transfer operations**
- **Part of Intel's Total System Architecture: MULTIBUS,® iLBX,™ MULTICHANNEL,™ and iSBX™**
- **Low-cost "vehicle" to incorporate the latest VLSI technology into iSBC®-based systems**
- **Provides increased functional capability and high performance**
- **Supported by a complete line of iSBC® base boards and iSBX™ MULTIMODULE™ boards, providing analog and digital I/O, high-speed math, serial and parallel I/O, video graphics, and peripheral controllers**

The iSBX™ I/O Expansion Bus is one of a family of standard bus structures resident within Intel's total system architecture. The iSBX bus is a modular, I/O expansion bus capable of increasing a single board computer's functional capability and overall performance by providing a structure to attach small iSBX MULTIMODULE™ boards to iSBC® base boards. It provides for rapid incorporation of new VLSI into iSBC MULTIBUS® systems, reducing the threat of system obsolescence. The iSBX bus offers users new economics in design by allowing both system size and system cost to be kept at minimum. As a result, the system design achieves maximum on-board performance while allowing the MULTIBUS interface to be used for other system activities. The iSBX bus enables users to add-on capability to a system as the application demands it by providing off-the-shelf standard MULTIMODULE boards in the areas of graphics controllers, advanced mathematics functions, parallel and serial I/O, disk and tape peripheral controllers, and magnetic bubble memory. A full line of MULTIBUS boards and iSBX MULTIMODULE boards are available from Intel and other third party sources in the industry.



OCTOBER 1984
ORDER NUMBER: 230893-002

FUNCTIONAL DESCRIPTION

Bus Elements

The iSBX™ MULTIMODULE™ system is made up of two basic elements: base boards and iSBX MULTIMODULE boards. In an iSBX system, the role of the base board is simple. It decodes I/O addresses and generates the chip selects for the iSBX MULTIMODULE boards.

The iSBX bus supports two classes of base boards, those with direct memory access (DMA) support and those without. Base boards with DMA support have DMA controllers that work in conjunction with an iSBX MULTIMODULE board (with DMA capability) to perform direct I/O to memory or memory to I/O operations. Base boards without DMA support use a subset of the iSBX bus and simply do not use the DMA feature of the iSBX MULTIMODULE board.

The iSBX MULTIMODULE boards are small, specialized, I/O mapped boards which plug into base boards. The iSBX boards connect to the iSBX bus connector and convert iSBX bus signals to a defined I/O interface.

Bus Interface/Signal Line Descriptions

The iSBX bus interface can be grouped into six functional classes: control lines, address and chip select lines, data lines, interrupt lines, option lines, and power lines. The iSBX bus provides nine control lines that define the communications protocol between base board and iSBX MULTIMODULE boards. These control lines are used to manage the general operation of the bus by specifying the type of transfer, the coordination of the transfer, and the overall state of the transfer between devices. The five address and chip select signal lines are used in conjunction with the commmand lines to establish the I/O port address being accessed, effectively selecting the proper iSBX MULTIMODULE. The data lines on the iSBX bus can number 8 or 16, and are used to transmit or receive information to or from the iSBX MULTIMODULE ports. Two interrupt lines are provided to make interrupt requests possible from the iSBX board to the base board. Two option lines are reserved on the bus for unique user requirements, while several power lines provide +5 and ±12 volts to the iSBX boards.

Bus Pin Assignments

The iSBX bus uses widely available, reliable connectors that are available in 18/36 pin for 8-bit devices and 22/44 pin for 16-bit devices. The male iSBX connector is attached to the iSBX MULTIMODULE board and the female iSBX connector is attached to the base board. Figure 2 shows the dimensions and pin numbering of the 18/36 pin iSBX connector, while Figure 3 does the same for the 22/44 pin iSBX connector. A unique scheme allows the 16-bit female connector to support 8 or 16-bit male MULTIMODULE boards. Table 1 lists the signal/pin assignments for the bus.

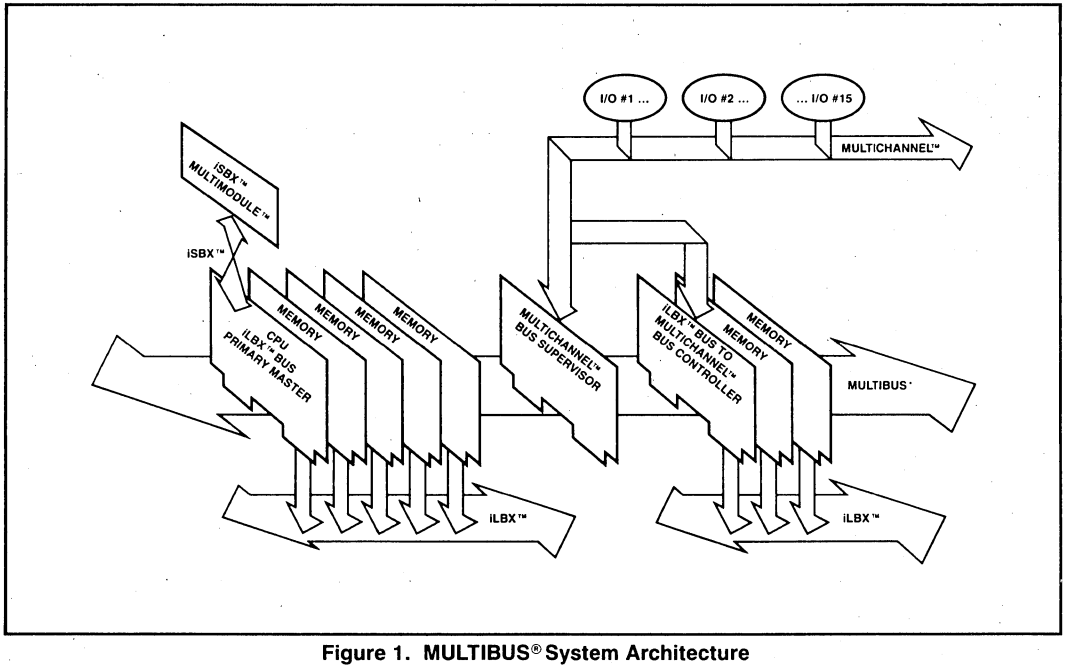


Figure 1. MULTIBUS® System Architecture

Table 1. iSBX™ Signal/Pin Assignments

Pin[1]	Menmonic	Description	Pin[1]	Mnemonic	Description
43	MD8	MDATA Bit 8	44	MD9	MDATA Bit 9
41	MDA	MDATA Bit A	42	MDB	MDATA Bit F
39	MDC	MDATA Bit C	40	MDD	MDATA Bit D
37	MDE	MDATA Bit E	38	MDF	MDATA Bit F
35	GND	Signal Gnd	36	+5V	+5 Volts
33	MD0	MDATA Bit 0	34	MDRQT	M DMA Request
31	MD1	MDATA Bit 1	32	MDACK/	M DMA Acknowledge
29	MD2	MDATA Bit 2	30	OPT0	Option 0
27	MD3	MDATA Bit 3	28	OPT1	Option 1
25	MD4	MDATA Bit 4	26	TDMA	Terminate DMA
23	MD5	MDATA Bit 5	24		Reserved
21	MD6	MDATA Bit 6	22	MCS0/	M Chip Select 0
19	MD7	MDATA Bit 7	20	MCS1/	M Chip Select 1
17	GND	Signal Gnd	18	+5V	+5 Volts
15	IORD/	I/O Read Cmd	16	MWAIT/	M Wait
13	IOWRT/	I/O Write Cmd	14	MINTR0	M Interrupt 0
11	MA0	M Address 0	12	MINTR1	M Interrupt 1
9	MA1	M Address 1	10		Reserved
7	MA2	M Address 2	8	MPST/	iSBX Multimodule Board Present
5	RESET	Reset	6	MCLK	M Clock
3	GND	Signal Gnd	4	+5V	+5 Volts
1	+12V	+12 Volts	2	-12V	-12 Volts

Notes:
1. Pins 37-44 are used only on 8/16-bit systems
2. All undefined pins are reserved for future use.

Bus Operation Protocol

COMMAND OPERATION

The iSBX bus supports two types of transfer operations between iSBX elements: I/O Read and I/O Write. An iSBX board can respond to these I/O transfers using either full speed mode or extended mode.

For a full speed I/O Read (Figure 4) the base board generates a valid I/O address and a valid chip select for the iSBX MULTIMODULE board. After set-up, the base board activates the I/O Read line causing the iSBX board to generate valid data from the addressed I/O port. The base board then reads the data and removes the read command, address, and chip select. The full speed I/O Write (Figure 5) operation is similar to the I/O Read except that the base board generates valid data on the lines and keeps the write command line active for the specified a hold time.

The extended Read operation (Figure 6) is used by iSBX MULTIMODULE boards that aren't configured to meet full speed specifications. It's operation is similar to full speed mode, but must use a wait signal to ensure proper

data transfer. The base board begins the operation by generating a valid I/O address and chip select. After set-up, the base board activates the Read line causing the iSBX board to generate a Wait signal. This causes the CPU on the base board to go into a wait state. When the iSBX board has placed valid Read data on the data lines, the MULTIMODULE board will remove the Wait signal and release the base board CPU to read the data and deactivate the command, address, and chip select. The extended Write operation (Figure 7) is similar to the extended Read except that the Wait signal is generated after the base board places valid Write data on the data lines. The iSBX board removes the Wait signal when the write pulse width requirements are satisfied, and the base board can then remove the write command after the hold time is met.

DMA OPERATION

An iSBX MULTIMODULE system can support DMA when the base board has a DMA controller and the iSBX MULTIMODULE board can support DMA mode. Burst mode DMA is fully supported, but for clarity and simplicity, only a single DMA transfer for an 8-bit base board is discussed.

A DMA cycle (Figure 8) is initiated by the iSBX board when it activates the DMA request line going to the DMA controller on the base board. When the DMA controller gains control of the base board bus, it acknowledges back to the iSBX board and activates an I/O or Memory Read. The DMA controller then activates an I/O or Memory Write respectively. The iSBX board removes the DMA request during the cycle to allow completion of the DMA cycle. Once the write operation is complete, the DMA controller is free to deactivate the write and read command lines after a data hold time.

INTERRUPT OPERATION

The iSBX MULTIMODULE board on the iSBX bus can support interrupt operations over its interrupt lines. The iSBX board initiates an interrupt by activating one of its two interrupt lines which connect to the base board. The CPU processes the interrupt and executes the interrupt service routine. The interrupt service routine signals the iSBX MULTIMODULE board to remove the interrupt, and then returns control to the main line program when the service routine is completed.

Please refer to the Intel iSBX Bus Specification for more detailed information on its operation and implementation.

SPECIFICATIONS

Word Size

Data — 8, 16-bit

Port Assignments

Power Supply Specifications

Table 3.

Minimum (volts)	Nominal (volts)	Maximum (volts)	Maximum (current)*
+4.75	+5.0	+5.25	3.0A
+11.4	+12	+12.6	1.0A
12.6	-12	-11.4	1.0A
—	GND	—	3.0A

* Per iSBX Multimodule board mounted on base board.

Table 2. iSBX™ MULTIMODULE™ Base Board Port Assignments

iSBX™ Connector Number	Chip Select	8-Bit Base Board Address	16-Bit Base Board Address (8-bit mode)	16-Bit Base Board Address (16-bit mode)
iSBX™ 1	MCS0/ MCS1/	F0-F7 F8-FF	0A0-0AF 0B0-0BF	0A0, 2, 4, 6, 8, A, C, E 0A1, 3, 5, 7, 9, B, D, F
iSBX™ 2	MCS0/ MCS1/	C0-C7 C8-CF	080-08F 090-09F	080, 2, 4, 6, 8, A, C, E 081, 3, 5, 7, 9, B, D, F
iSBX™ 3	MCS0/ MCS1/	B0-B7 B8-BF	060-06F 060-06F	060, 2, 4, 6, 8, A, C, E 061, 3, 5, 7, 9, B, D, F

DC Specifications

Table 4. iSBX™ MULTIMODULE™ Board I/O DC Specifications

Output[1]

Bus Signal Name	Type[2] Drive	I_{OL} Max Min (mA)	@ Volts (V_{OL} Max)	I_{OH} Max Min (μA)	@ Volts (V_{OH} Min)	C_O (Min) (pf)
MD0-MDF	TRI	1.6	0.5	−200	2.4	130
MINTR0-1	TTL	2.0	0.5	−100	2.4	40
MDRQT	TTL	1.6	0.5	− 50	2.4	40
MWAIT/	TTL	1.6	0.5	− 50	2.4	40
OPT1-2	TTL	1.6	0.5	− 50	2.4	40
MPST/	TTL	Note 3				

Input[1]

Bus Signal Name	Type[2] Receiver	I_{IL} Max (mA)	@ V_{IN} MAX (volts) Test Cond.	I_{IH} Max (μA)	@ V_{IN} MAX (volts) Test Cond.	C_I Max (pf)
MD0-MDF	TRI	−0.5	0.4	70	2.4	40
MA0-MA2	TTL	−0.5	0.4	70	2.4	40
MCS0/-MCS1/	TTL	−4.0	0.4	100	2.4	40
MRESET	TTL	−2.1	0.4	100	2.4	40
MDACK/	TTL	−1.0	0.4	100	2.4	40
IORD/ IOWRT/	TTL	−1.0	0.4	100	2.4	40
MCLK	TTL	−2.0	0.4	100	2.4	40
OPT1-OPT2	TTL	−2.0	0.4	100	2.4	40

NOTES:
1. Per iSBX Multimodule I/O board.
2. TTL = standard totem pole output. TRI = Three-state.
3. iSBX Multimodule board must connect this signal to ground.

All Inputs: Max V_{IL} = 0.8V
Min V_{IH} = 2.0V

Connectors

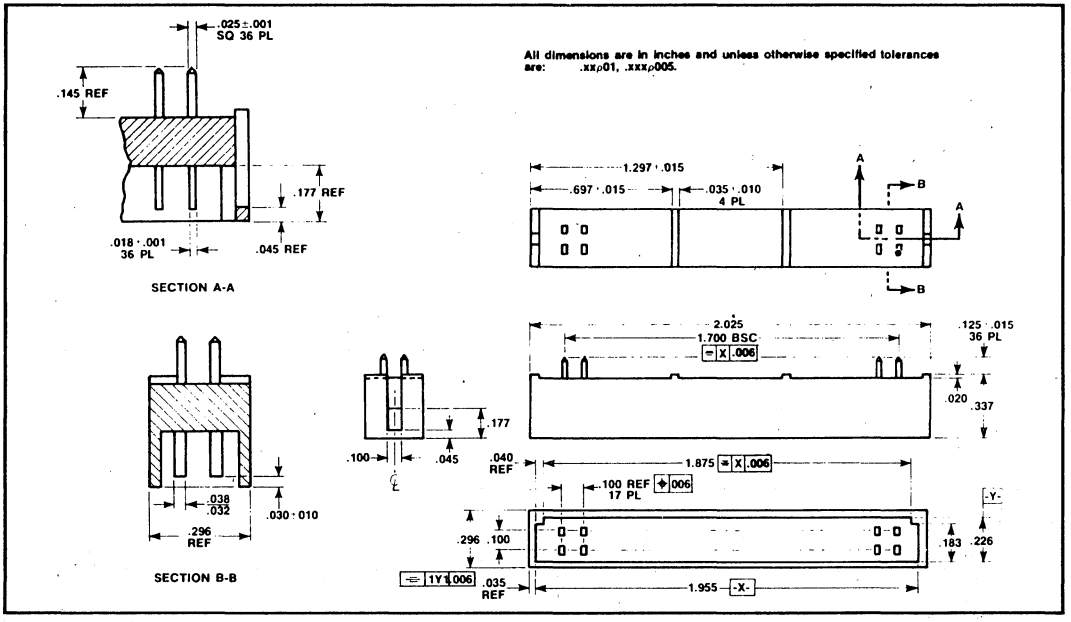


Figure 2. 18/36 Pin iSBX™ Connector

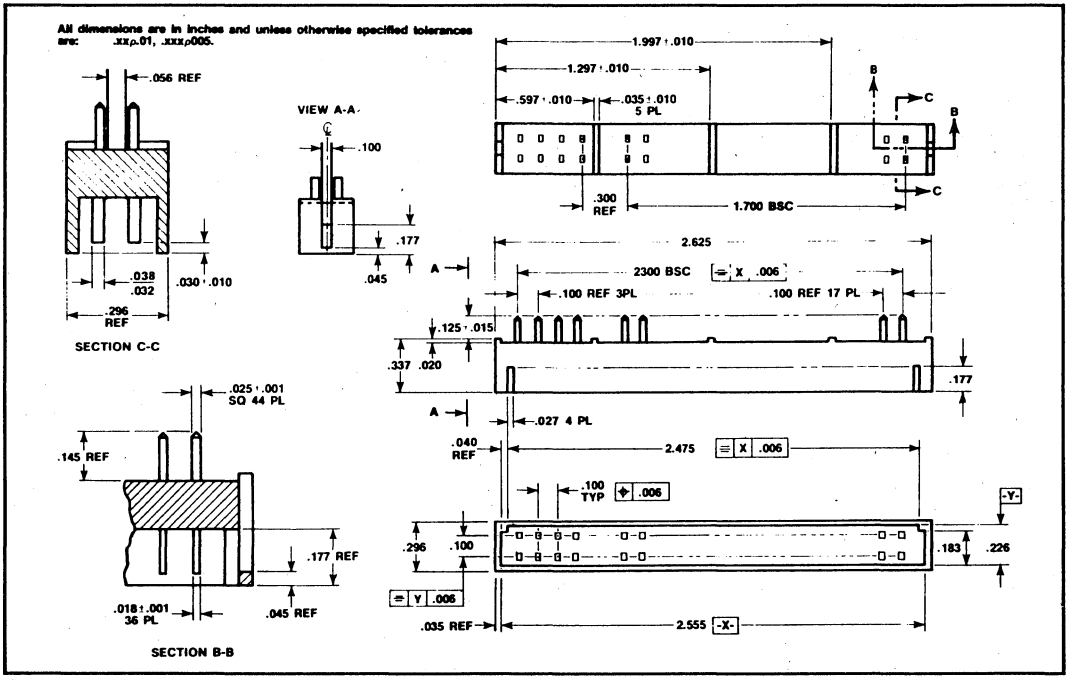


Figure 3. 22/44 Pin iSBX™ Connector

Bus Timing Diagrams

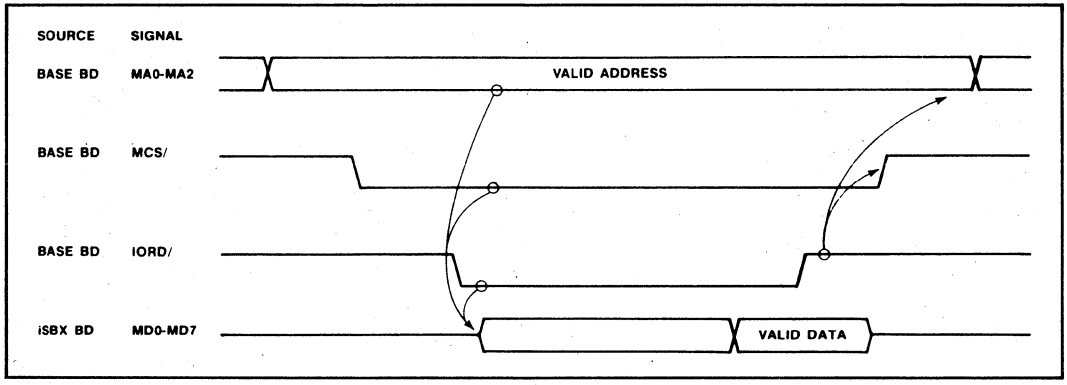


Figure 4. iSBX™ MULTIMODULE™ Read, Full Speed

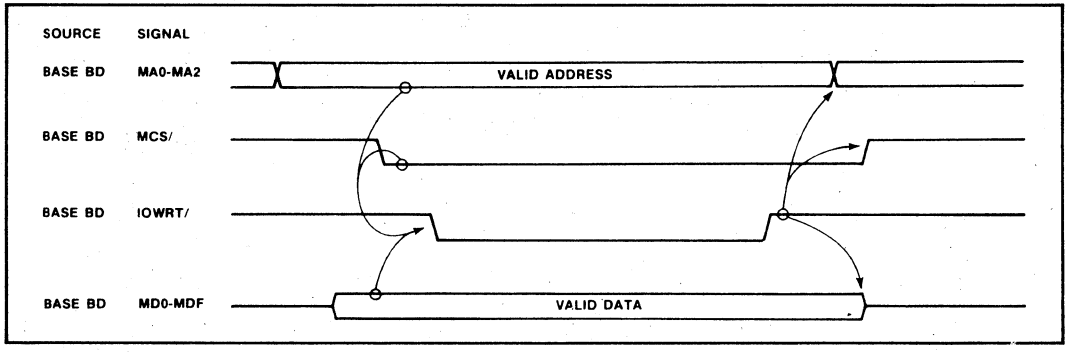


Figure 5. iSBX™ MULTIMODULE™ Board Write, Full Speed

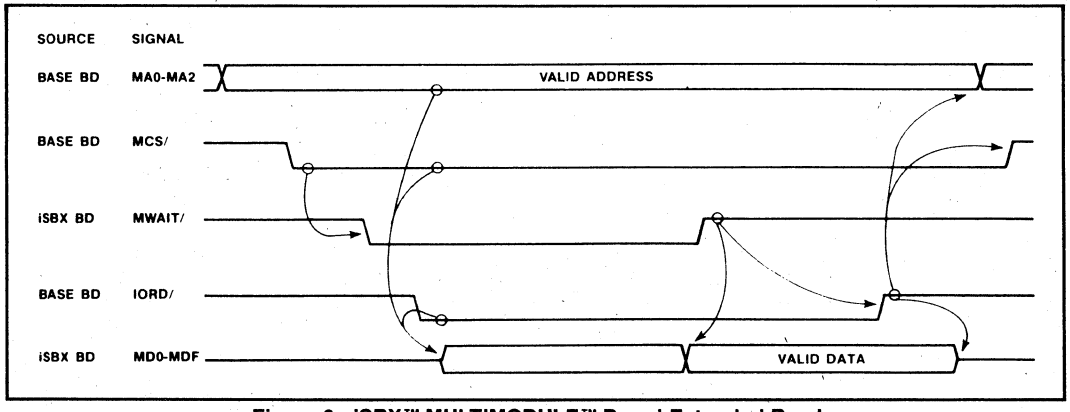


Figure 6. iSBX™ MULTIMODULE™ Board Extended Read

Bus Timing Diagram (Con't)

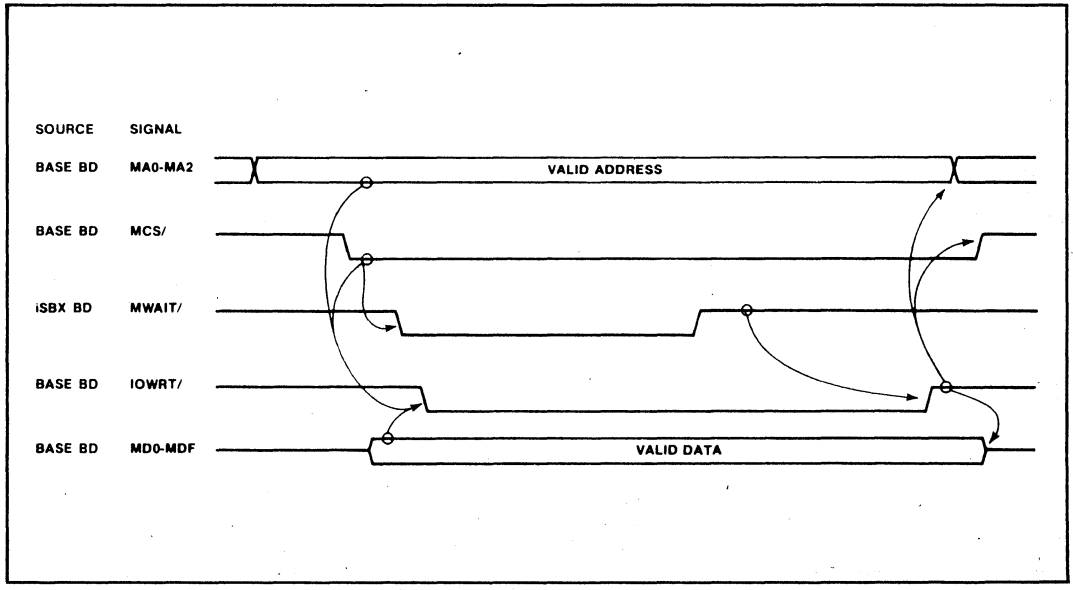


Figure 7. iSBC™ MULTIMODULE™ Board Extended Write

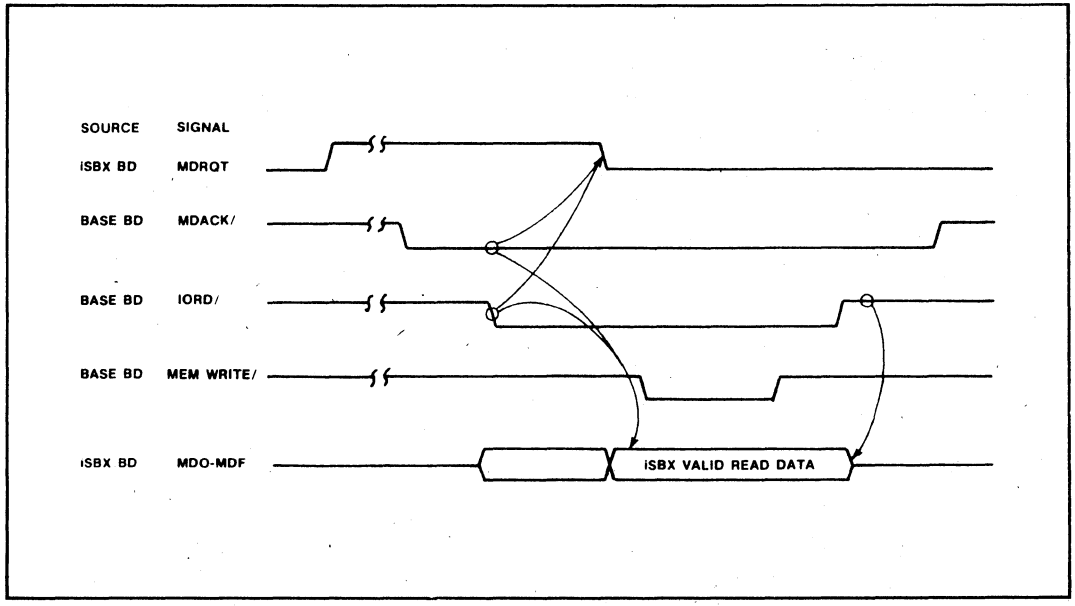


Figure 8. iSBX™ MULTIMODULE™ Board DMA Cycle (iSBX™ MULTIMODULE™ to Base Board Memory)

Board Outlines

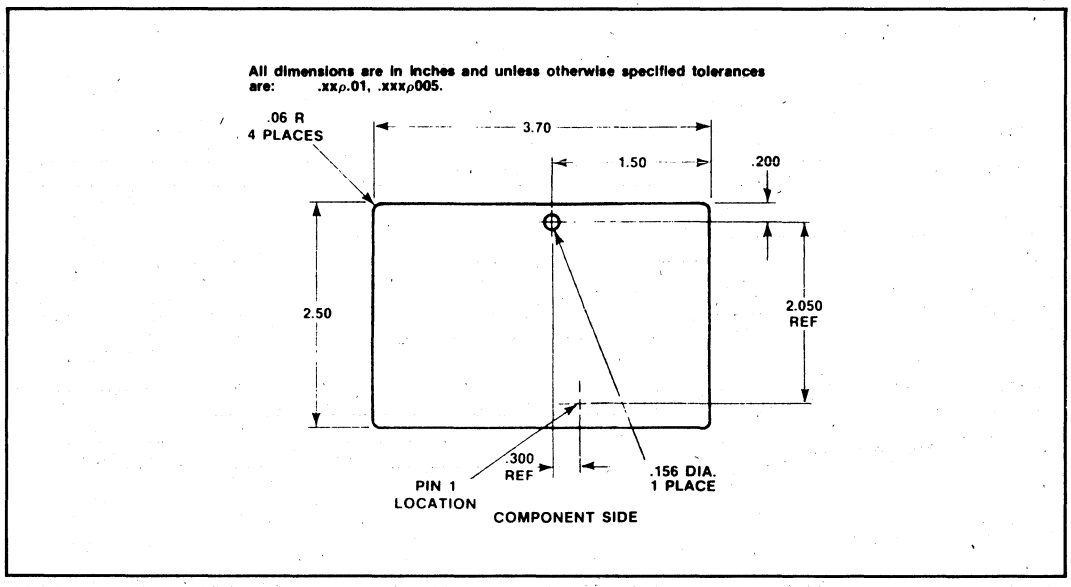


Figure 9. iSBX™ Board Outline

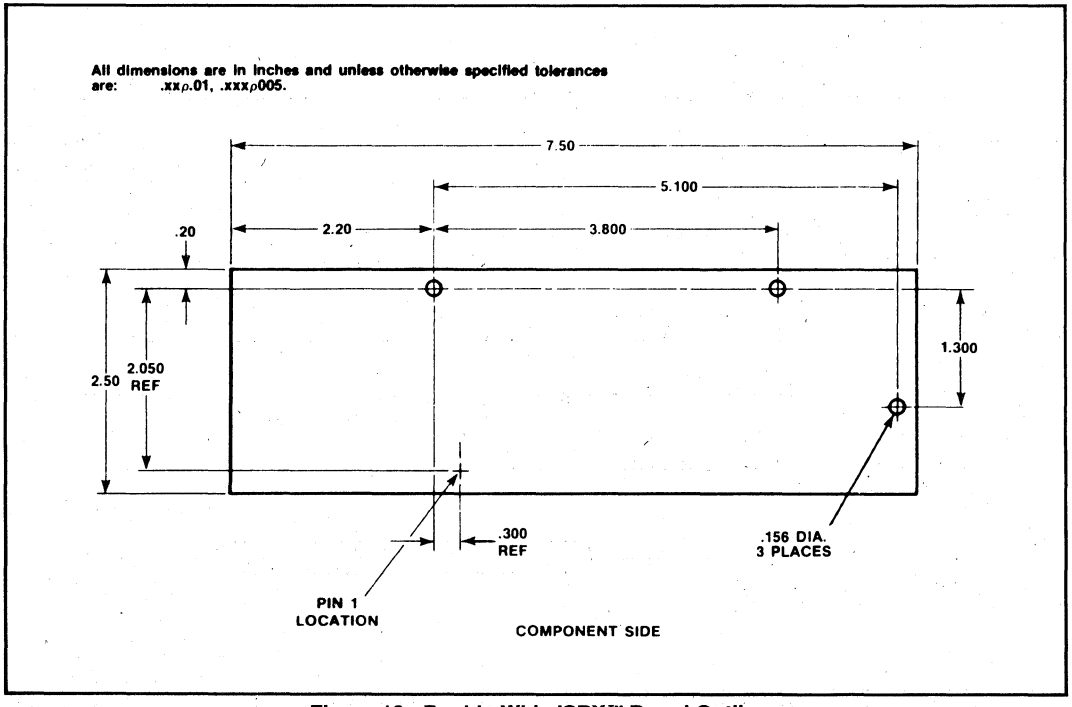


Figure 10. Double Wide iSBX™ Board Outline

Environmental Characteristics

Operating Temperature — 0 to 55°C

Humidity — 90% maximum relative; non-condensing

Reference Manuals

210883 — MULTIBUS Handbook

intel®

APPLICATION NOTE

AP-28A

January, 1983

Intel MULTIBUS® Interfacing

Joe Barthmaier

Order Number: 210895-001

I. INTRODUCTION

A significant measure of the power and flexibility of the Intel OEM Computer Product Line can be attributed to the design of the Intel MULTIBUS system bus. The bus structure provides a common element for communication between a wide variety of system modules which include: Single Board Computers, memory, digital, and analog I/O expansion boards, and peripheral controllers.

The purpose of this application note is to help you develop a working knowledge of the Intel MULTIBUS specification. This knowledge is essential for configuring a system containing multiple modules. Another purpose is to provide you with the information necessary to design a bus interface for a slave module. One of the tools that will be used to achieve this goal is the complete description of a MULTIBUS slave design example. Other portions of this application note provide an in depth examination of the bus signals, operating characteristics, and bus interface circuits.

This application note was originally written in 1977. Since 1977, the MULTIBUS specification has been significantly expanded to cover operation with both 8 and 16-bit system modules and with an auxiliary power bus. This application note now contains information on these new MULTIBUS specification features.

In addition, a detailed MULTIBUS specification has also been published which provides the user with further information concerning MULTIBUS interfacing. The MULTIBUS specification and other useful documents are listed in the overleaf of this note under Related Intel Publications.

II. MULTIBUS® SYSTEM BUS DESCRIPTION

Overview

The Intel MULTIBUS signal lines can be grouped in the following categories: 20 address lines, 16 bidirectional data lines, 8 multilevel interrupt lines, and several bus control, timing and power supply lines. The address and data lines are driven by three-state devices, while the interrupt and some other control lines are open-collector driven.

Modules that use the MULTIBUS system bus have a master-slave relationship. A bus master module can drive the command and address lines: it can control the bus. A Single Board Computer is an example of a bus master. A bus slave cannot control the bus. Memory and I/O expansion boards are examples of bus slaves. The MULTIBUS architecture provides for both 8 and 16-bit bus masters and slaves.

Notice that a system may have a number of bus masters. Bus arbitration results when more than one master requests control of the bus at the same time. A bus clock is usually provided by one of the bus masters and may be derived independently from the processor clock. The bus clock provides a timing reference for resolving bus contention among multiple requests from bus masters. For example, a processor and a DMA (direct memory access) module may both request control of the bus. This feature allows different speed masters to share resources on the same bus. Actual transfers via the bus, however, proceed asynchronously with respect to the bus clock. Thus, the transfer speed is dependent on the transmitting and receiving devices only. The bus design prevents slow master modules from being handicapped in their attempts to gain control of the bus, but does not restrict the speed at which faster modules can transfer data via the same bus. Once a bus request is granted, single or multiple read/write transfers can proceed. The most obvious applications for the master-slave capabilities of the bus are multiprocessor configurations and high-speed direct-memory-access (DMA) operations. However, the master-slave capabilities of the bus are by no means limited to these two applications.

MULTIBUS® Signal Descriptions

This section defines the signal lines that comprise the Intel MULTIBUS system bus. These signals are contained on either the P1 or P2 connector of boards compatible with the MULTIBUS specification. The P1 signal lines contain the address, data, bus control, bus exchange, interrupt and power supply lines. The P2 signal lines contain the optional auxiliary signal lines. Most signals on the bus are active-low. For example, a low level on a control signal on the bus indicates active, while a low level on an address or data signal on the bus represents logic "1" value.

NOTE

In this application note, a signal will be designated active-low by placing a slash (/) after the mnemonic for the signal.

Appendix A contains a pin assignment list of the following signals:

MULTIBUS P1 Signal Lines —

Initialization Signal Line

INIT/

Initialization signal; resets the entire system to a known internal state. INIT/ may be driven by one of the bus masters or by an external source such as a front panel reset switch.

Address and Inhibit Lines

ADR0/ - ADR13/

20 address lines; used to transmit the address of the memory location or I/O port to be accessed. The lines are labeled ADR0/ through ADR9/, ADRA/ through ADRF/ and ADR10/ through ADR13/. ADR13/ is the most significant bit. 8-bit masters use 16 address lines (ADR0/ - ADRF/) for memory addressing and 8 address lines (ADR0/ - ADR7/) for I/O port selection. 16-bit masters use all twenty address lines for memory addressing and 12 address lines (ADR0/ - ADRB/) for I/O port selection. Thus, 8-bit masters may address 64K bytes of memory and 256 I/O devices while 16-bit masters may address 1 megabyte of memory and 4096 I/O devices. (The 8086 CPU actually permits 16 address bits to be used to specify I/O devices, the MULTIBUS specification, however, states that only the low order 12 address bits can be used to specify I/O ports.) In a 16-bit system, the ADR0/ line is used to indicate whether a low (even) byte or a high (odd) byte of memory or I/O space is being accessed in a word oriented memory or I/O device.

BHEN/

Byte High Enable; the address control line which is used to specify that data will be transferred on the high byte (DAT8/ - DATF/) of the MULTIBUS data lines. With current iSBC boards, this signal effectively specifies that a word (two byte) transfer is to be performed. This signal is used only in systems which incorporate sixteen bit memory or I/O modules.

INH1/

Inhibit RAM signal; prevents RAM memory devices from responding to the memory address on the system address bus. INH1/ effectively allows ROM memory devices to override RAM devices when ROM and RAM memory are assigned the same memory addresses. INH1/ may also be used to allow memory mapped I/O devices to override RAM memory.

INH2/

Inhibit ROM signal; prevents ROM memory devices from responding to the memory address on the system address bus. INH2/ effectively allows auxiliary ROM (e.g., a bootstrap program) to override ROM devices when ROM and auxiliary ROM memory are assigned the same memory addresses. INH2/ may also be used to allow memory mapped I/O devices to override ROM memory.

Data Lines

DAT0/ - DATF/

16 bidirectional data lines; used to transmit or receive information to or from a memory location or I/O port. DATF/ being the most significant bit. In 8-bit systems, only lines DAT0/ - DAT7/ are used (DAT7/ being the most significant bit). In 16-bit systems, either 8 or 16 lines may be used for data transmission.

Bus Priority Resolution Lines

BCLK/

Bus clock; the negative edge (high to low) of BCLK/ is used to synchronize bus priority resolution circuits. BCLK/ is asynchronous to the CPU clock. It has a 100 ns minimum period and a 35% to 65% duty cycle. BCLK/ may be slowed, stopped, or single stepped for debugging.

CCLK/

Constant clock; a bus signal which provides a clock signal of constant frequency for unspecified general use by modules on the system bus. CCLK/ has a minimum period of 100 ns and a 35% to 65% duty cycle.

BPRN/

Bus priority in signal; indicates to a particular master module that no higher priority module is requesting use of the system bus. BPRN/ is synchronized with BCLK/. This signal is not bused on the backplane.

BPRO/

Bus priority out signal; used with serial (daisy chain) bus priority resolution schemes. BPRO/ is passed to the BPRN/ input of the master module with the next lower bus priority. BPRO/ is synchronized with BCLK/. This signal is not bused on the backplane.

BUSY/

Bus busy signal; an open collector line driven by the bus master currently in control to indicate that the bus is currently in use. BUSY/ prevents all other master modules from gaining control of the bus. BUSY/ is synchronized with BCLK/.

BREQ/

Bus request signal; used with a parallel bus priority network to indicate that a particular master module requires use of the bus for one or more data transfers. BREQ/ is synchronized with BCLK/. This signal is not bused on the backplane.

CBRQ/

Common bus request; an open-collector line which is driven by all potential bus masters and is used to inform the current bus master that another master wishes to use the bus. If CBRQ/ is high, it indicates to the bus master that no other master is requesting the bus, and therefore, the present bus master can retain the bus. This saves the bus exchange overhead for the current master.

Information Transfer Protocol Lines

A bus master provides separate read/write command signals for memory and I/O devices: MRDC/, MWTC/, IORC/ and IOWC/, as explained below. When a read/write command is active, the address signals must be stabilized at all slaves on the bus. For this reason, the protocol requires that a bus master must issue address signals (and data signals for a write operation) at least 50 ns ahead of issuing a read/write command to the bus, initiating the data transfer. The bus master must keep address signals unchanged until at least 50 ns after the read/write command is turned off, terminating the data transfer.

A bus slave must provide an acknowledge signal to the bus master in response to a read or write command signal.

MRDC/

Memory read command; indicates that the address of a memory location has been placed on the system address lines and specifies that the contents (8 or 16 bits) of the addressed location are to be read and placed on the system data bus. MRDC/ is asynchronous with respect to BCLK/.

MWTC/

Memory write command; indicates that the address of a memory location has been placed on the system address lines and that data (8 or 16 bits) has been placed on the system data bus. MWTC/ specifies that the data is to be written into the addressed memory location. MWTC/ is asynchronous with respect to BCLK/.

IORC/

I/O read command; indicates that the address of an input port has been placed on the system address bus and that the data (8 or 16 bits) at that input port is to be read and placed on the system data bus. IORC/ is asynchronous with respect to BCLK/.

IOWC/

I/O write command; indicates that the address of an output port has been placed on the system address bus and that the contents of the system data bus (8 or 16 bits) are to be output to the address port. IOWC/ is asynchronous with respect to BCLK/.

XACK/

Transfer acknowledge signal; the required response of a slave board which indicates that the specified read/write operation has been completed. That is, data has been placed on, or accepted from, the system data bus lines. XACK/ is asynchronous with respect to BCLK/.

Asynchronous Interrupt Lines

INT0/ - INT7/

8 Multi-level, parallel interrupt request lines;

used with a parallel interrupt resolution network. INT0/ has the highest priority, while INT7/ has lowest priority. Interrupt lines should be driven with open collector drivers.

INTA/

Interrupt acknowledge; an interrupt acknowledge line (INTA/), driven by the bus master, requests the transfer of interrupt information onto the bus from slave priority interrupt controllers (8259s or 8259As). The specific information timed onto the bus depends upon the implementation of the interrupt scheme. In general, the leading edge of INTA/ indicates that the address bus is active while the trailing edge indicates that data is present on the data lines.

MULTIBUS P2 Signal Lines — The signals contained on the MULTIBUS P2 auxiliary connector are used primarily by optional power back-up circuitry for memory protection. P2 signals are not bused on the backplane, and therefore, require a separate connector for each board using the P2 signals. Present iSBC boards have a slot in the card edge and should be used with a keyed P2 edge connector. Use of the P2 signal lines is optional.

ACLO

AC Low; this signal generated by the power supply goes high when the AC line voltage drops below a certain voltage (e.g., 103v AC in 115v AC line voltage systems) indicating D.C. power will fail in 3 msec. ACLO goes low when all D.C. voltages return to approximately 95% of the regulated value. This line must be pulled up by the optional standby power source, if one is used.

PFIN/

Power fail interrupt; this signal interrupts the processor when a power failure occurs, it is driven by external power fail circuitry.

PFSN/

Power fail sense; this line is the output of a latch which indicates that a power failure has occurred. It is reset by PFSR/. The power fail sense latch is part of external power fail circuitry and must be powered by the standby power source.

PFSR/

Power fail sense reset; this line is used to reset the power fail sense latch (PFSN/).

MPRO/

Memory protect; prevents memory operation during period of uncertain DC power, by inhibiting memory requests. MPRO/ is driven by external power fail circuitry.

ALE

Address latch enable; generated by the CPU (8085 or 8086) to provide an auxiliary address latch.

HALT/

Halt; indicates that the master CPU is halted.

AUX RESET/

Auxiliary Reset; this externally generated signal initiates a power-up sequence.

WAIT/

Bus master wait state; this signal indicates that the processor is in a wait state.

Reserved — Several P1 and P2 connector bus pins are unused. However, they should be regarded as reserved for dedicated use in future Intel products.

Power Supplies — The power supply bus pins are detailed in Appendix A which contains the pin assignment of signals on the MULTIBUS backplane.

It is the designer's responsibility to provide adequate bulk decoupling on the board to avoid current surges on the power supply lines. It is also recommended that you provide high frequency

decoupling for the logic on your board. Values of 22uF for +5v and +12v pins and 10uF for -5v and -12v pins are typical on iSBC boards.

Operating Characteristics

Beyond the definition of the MULTIBUS signals themselves, it is important to examine the operating characteristics of the bus. The AC requirements outline the timing of the bus signals and in particular, define the relationships between the various bus signals. On the other hand, the DC requirements specify the bus driver characteristics, maximum bus loading per board, and the pull-up/down resistors.

The AC requirements are best presented by a discussion of the relevant timing diagrams. Appendix B contains a list of the MULTIBUS timing specifications. The following sections will discuss data transfers, inhibit operations, interrupt operations, MULTIBUS multi-master operation and power fail considerations.

Data Transfers — Data transfers on the MULTIBUS system bus occur with a maximum bandwidth of 5 MHz for single or multiple read/write transfers. Due to bus arbitration and memory access time, a typical maximum transfer rate is often on the order of 2 MHz.

Read Data

Figure 1 shows the read operation AC timing diagram. The address must be stable (t_{AS}) for a minimum of 50 ns before command (IORC/ or MRDC/). This time is typically used by the bus interface to decode the address and thus provide the required device selects. The device selects establish the data paths on the user system in anticipation of the strobe signal (command) which will follow. The minimum command pulse width is 100 ns. The address must remain stable for at least 50 ns following the command (t_{AH}). Valid data should not be driven onto the bus prior to command, and must not be removed until the command is cleared. The XACK/ signal, which is a response indicating the specified read/write operation has been completed, must coincide or follow both the read access and valid data (t_{DXL}). XACK/ must be held until the command is cleared (t_{XAH}).

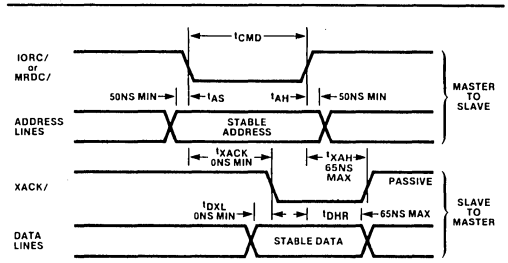


Figure 1. Read AC Timing

Write Data

The write operation AC timing diagram is shown in Figure 2. During a write data transfer, valid data must be presented simultaneously with a stable address. Thus, the write data setup time (t_{DS}) has the same requirement as the address setup time (t_{AS}). The requirement for stable data both before and after command (IOWC/ or MWTC/) enables the bus interface circuitry to latch data on either the leading or trailing edge of command.

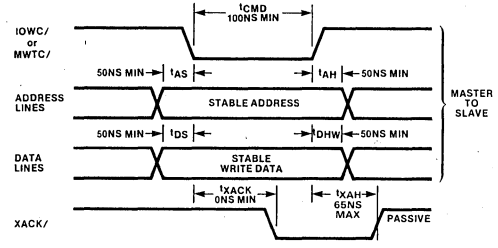


Figure 2. Write AC Timing

Data Byte Swapping in 16-bit Systems

A 16-bit master may transfer data on the MULTIBUS data lines using 8-bit or 16-bit paths depending on whether a byte or word (2 byte) operation has been specified. (A word transfer specified with an odd I/O or memory address will actually be executed as two single byte transfers.) An 8-bit master may only perform byte transfers on the MULTIBUS data lines DAT0/ - DAT7/.

In order to maintain compatibility with older 8-bit masters and slaves, a byte swapping buffer is included in all new 16-bit masters and 16-bit slaves. In the iSBC product line, all byte transfers will take place on the low 8 data lines DAT0/ - DAT7/. Figure 3 contains a example of 8/16-bit

data driver logic for 16-bit master and slave systems. In the 8/16-bit system, there are three sets of buffers; the lower byte buffer which accesses DAT0/ - DAT7/, the upper byte buffer which accesses DAT8/ - DATF/, and the swap byte buffer which accesses the MULTIBUS data lines DAT0/ - DAT7/ and transfers the data to/from the on-board data bus lines D8 - DF.

Figure 4 summarizes the 8 and 16-bit data paths used for three types of MULTIBUS transfers. Two signals control the data transfers.

Byte High Enable (BHEN/) active indicates that the bus is operating in sixteen bit mode, and Address Bit 0 (ADR0/) defines an even or odd byte transfer address.

On the first type of transfer, BHEN/ is inactive, and ADR0/ is inactive indicating the transfer of an even eight bit byte. The transfer takes place across data lines DAT0/ - DAT7/.

On the second type of transfer, BHEN/ is inactive, and ADR0/ is active indicating the transfer of a high (odd) byte. On this type of transfer, the odd (high) byte is transferred through the Swap Byte Buffer to DAT0/ - DAT7/. This makes eight bit and sixteen bit systems compatible.

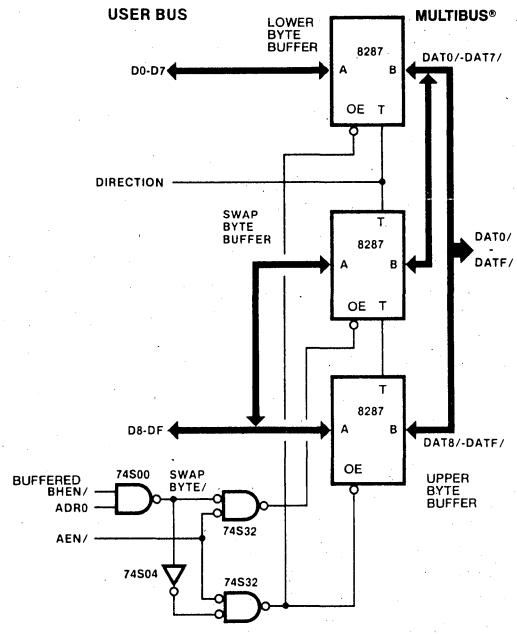


Figure 3. 8/16-Bit Data Drivers

16-BIT DEVICE / MULTIBUS®	BHEN/	ADR0/	MULTIBUS® TRANSFER DATA PATH	DEVICE BYTE TRANSFERRED
LOW, EVEN BYTES → DAT0/ - DAT7/; HIGH,ODD BYTES → DAT8/ - DATF/	H	H	8-BIT, DAT0/ - DAT7/	EVEN
LOW, EVEN BYTES → DAT0/ - DAT7/; HIGH,ODD BYTES → DAT8/ - DATF/	H	L	8-BIT, DAT0/ - DAT7/	ODD
LOW, EVEN BYTES → DAT0/ - DAT7/; HIGH, ODD BYTES → DAT8/ - DATF/	L	H	16-BIT, DAT0/ - DATF/	EVEN AND ODD

Figure 4. 8/16-Bit Device Transfer Operation

The third type of transfer is a 16 bit (word) transfer. This is indicated by BHEN/ being active, and ADR0/ being inactive. On this type of transfer, the low (even) byte is transferred on DAT0/ - DAT7/ and the high (odd) byte is transferred on DAT8/ - DATF/.

Note that the condition when both BHEN/ and ADR0/ are active is not used with present iSBC boards. This condition could be used to transfer a high odd byte of data on DAT8/ - DATF/, thus eliminating the need for the swap byte buffer. However, this is not a recommended transfer type, because it eliminates the capability of communicating with 8-bit modules.

Inhibit Operations — Bus inhibit operations are required by certain bootstrap and memory mapped I/O configurations. The purpose of the inhibit operation is to allow a combination of RAM, ROM, or memory mapped I/O to occupy the same memory address space. In the case of a bootstrap, it may be desirable to have both ROM and RAM memory occupy the same address space, selecting ROM instead of RAM for low order memory only when the system is reset. A system designed to use memory mapped I/O, which has actual memory occupying the memory mapped I/O address space, may need to inhibit RAM or ROM memory to perform its functions.

There are two essential requirements for a successful inhibit operation. The first is that the inhibit signal must be asserted as soon as possible, within a maximum of 100 ns (t_{CI}), after stable address. The second requirement for a successful inhibit operation is that the acknowledge must be delayed (t_{XACKB}) to allow the inhibited slave to terminate any irreversible timing operations initiated by detection of a valid command prior to its inhibit.

This situation may arise because a command can be asserted within 50 ns after stable address (t_{AS}) and yet inhibit is not required until 100 ns (t_{ID}) after stable address. The acknowledge delay time (t_{XACKB}) is a function of the cycle time of the inhibited slave memory. Inhibiting the iSBC 016 RAM board, for example, requires a minimum of 1.5 usec. Less time is typically needed to inhibit other memory modules. For example, the iSBC 104 board requires 475 ns.

Figure 5 depicts a situation in which both RAM

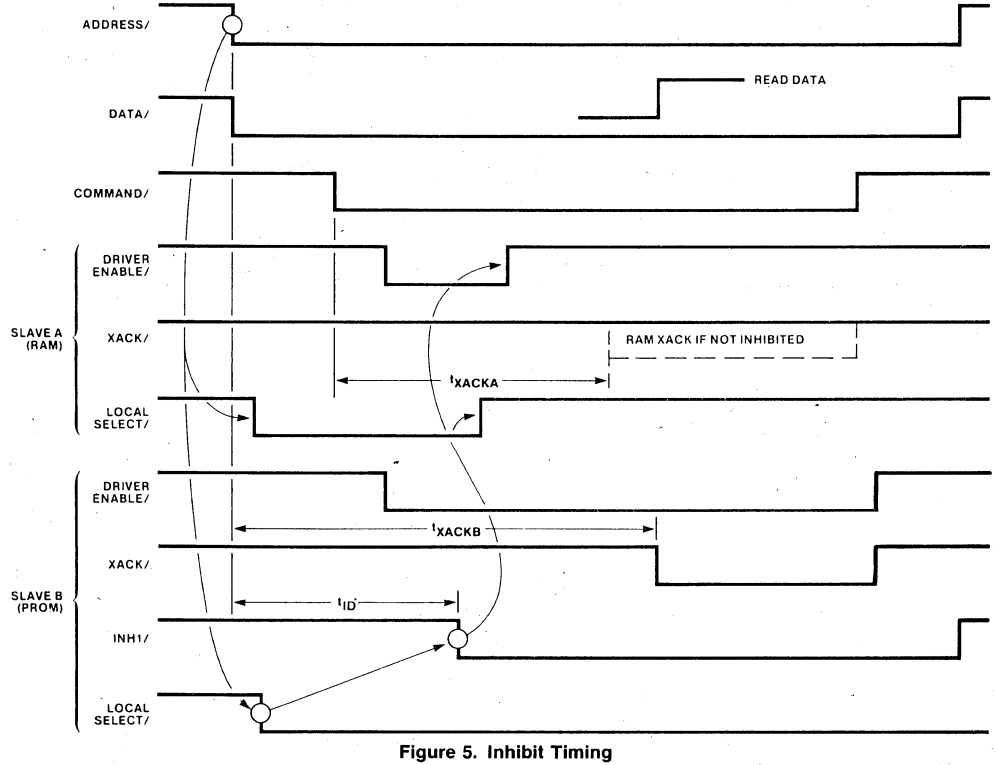


Figure 5. Inhibit Timing

and PROM memory have the same memory addresses. In this case, PROM inhibits RAM, producing the effect of PROM overriding RAM. After address is stable, local selects are generated for both the PROM and the RAM. The PROM local select produces the INH1/ signal which then removes the RAM local select and its driver enable. Because the slave RAM has been inhibited after it had already begun its cycle, the PROM XACK/ must be delayed (t_{XACKB}) until after the latest possible acknowledgement from the RAM (t_{XACKA}).

Interrupt Operations — The MULTIBUS interrupt lines INT0/ - INT7/ are used by a MULTIBUS master to receive interrupts from bus slaves, other bus masters or external logic such as power fail logic. A bus master may also contain internal interrupt sources which do not require the bus interrupt lines to interrupt the master. There are two interrupt implementation schemes used by bus interrupts, Non Bus Vectored Interrupts and Bus Vectored Interrupts. Non Bus Vectored Interrupts do not convey interrupt vector address information on the bus. Bus Vectored Interrupts are interrupts from slave Priority Interrupt Controllers (PICs) which do convey interrupt vector address information on the bus.

Non Bus Vectored Interrupts

Non Bus Vectored Interrupts are those interrupts whose interrupt vector address is generated by the bus master and do not require the MULTIBUS address lines for transfer of the interrupt vector address. The interrupt vector address is generated by the interrupt controller on the master and transferred to the processor over the local bus. The source of the interrupt can be on the master module or on other bus modules, in which case the bus modules use the MULTIBUS interrupt request lines (INT0/ - INT7/) to generate their interrupt requests to the bus master. When an interrupt request line is activated, the bus master performs it own interrupt operation and processes the interrupt. Figure 6 shows an example of Non Bus Vectored Interrupt implementation.

Bus Vectored Interrupts

Bus Vectored Interrupts (Figure 7) are those interrupts which transfer the interrupt vector address along the MULTIBUS address lines from the slave to the bus master using the INTA/ command signal for synchronization.

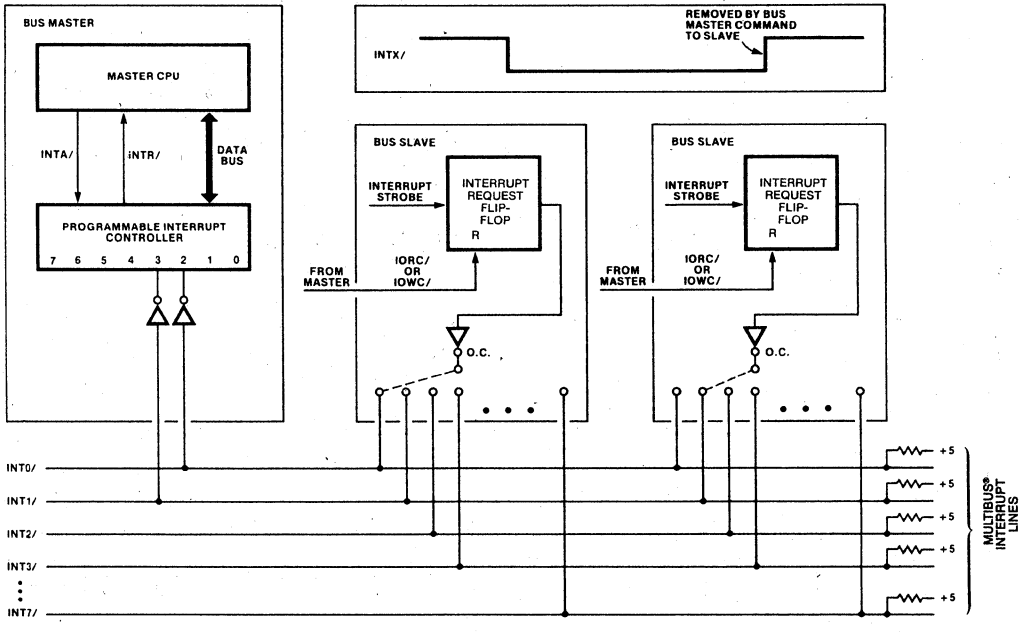


Figure 6. Non Bus Vectored Interrupt Implementation

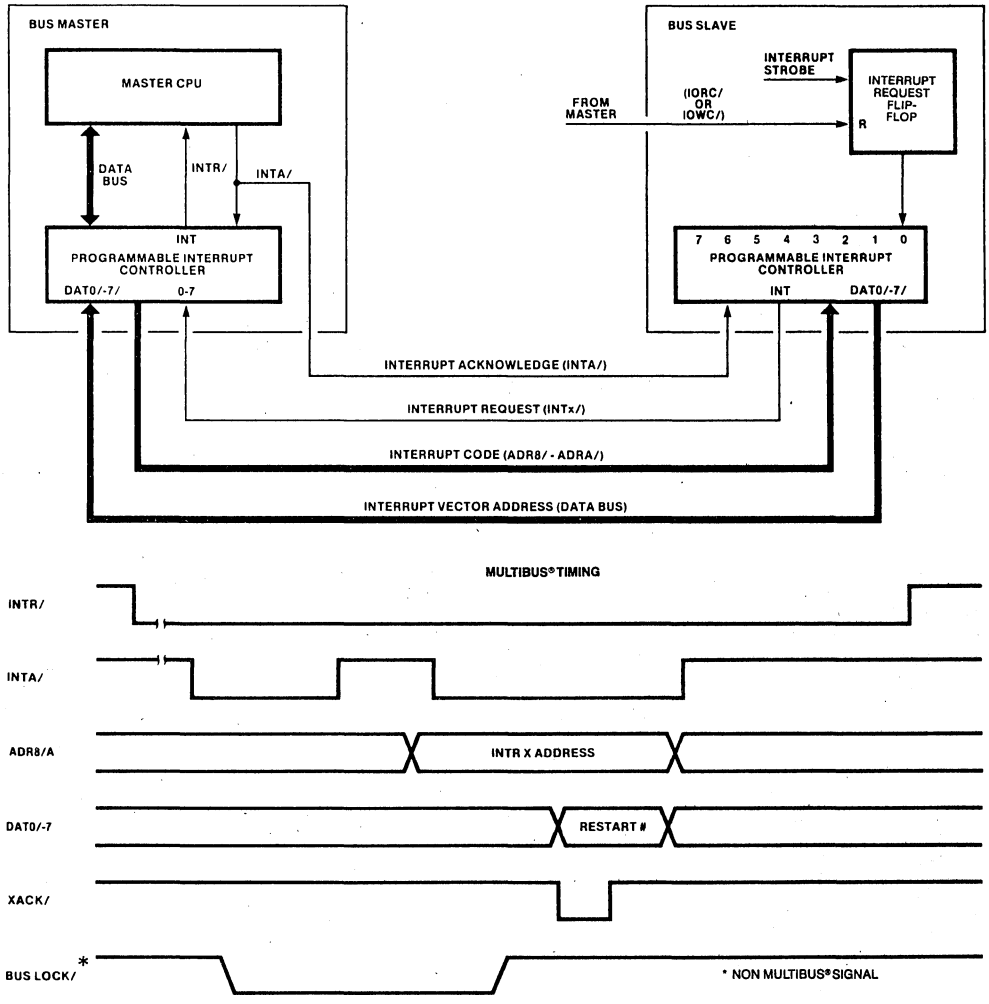


Figure 7. Bus Vectored Interrupt Logic (With 2 INTA/ Timing Diagram)

When an interrupt request from the MULTIBUS interrupt lines INT0/ - INT7/ occurs, the interrupt control logic on the bus master interrupts its processor. The processor on the bus master generates an INTA/ command which freezes the state of the interrupt logic on the MULTIBUS slaves for priority resolution. The bus master also locks (retains the bus between bus cycles) the MULTIBUS control lines to guarantee itself consecutive bus cycles. After the first INTA/ command, the bus master's interrupt control logic puts an interrupt code on to the MULTIBUS address lines ADR8/ - ADRA/. The interrupt code is the address of the highest priority active interrupt request line. At this point in the Bus Vectored Interrupt procedure, two different sequences could take place. The difference occurs, because the MULTIBUS specification can support masters which generate one additional INTA/ (8086 masters) or two additional INTA/s (8080A and 8085 masters).

If the bus master generates one additional INTA/, this second INTA/ causes the bus slave interrupt control logic to transmit an interrupt vector 8-bit pointer on the MULTIBUS data lines. The vector pointer is used by the bus master to determine the memory address of the interrupt service routine.

If the bus master generates two additional INTA/s, these two INTA/ commands allow the

bus slave to put a two byte interrupt vector address on to the MULTIBUS data lines (one byte for each INTA/). The interrupt vector address is used by the bus master to service the interrupt.

The MULTIBUS specification provides for only one type of Bus Vectored Interrupt operation in a given system. Slave boards which have an 8259 interrupt controller are only capable of 3 INTA/ operation (2 additional INTA/s after the first INTA/). Slave boards with the 8259A interrupt controller are capable of either 2 INTA/ or 3 INTA/ operation. All slave boards in a given system must operate in the same way (2 INTA/s or 3 INTA/s) if Bus Vectored Interrupts are to be used. However, the MULTIBUS specification does provide for Bus Vectored Interrupts and Non Bus Vectored Interrupts in the same system.

MULTIBUS® Multi-Master Operation — The MULTIBUS system bus can accommodate several bus masters on the same system, each one taking control of the bus as it needs to affect data transfers. The bus masters request bus control through a bus exchange sequence.

Two bus exchange priority resolution techniques are discussed, a serial technique and a parallel technique. Figures 8 and 9 illustrate these two techniques. The bus exchange operation discussed later is the same for both techniques.

Serial Priority Technique

Serial priority resolution is accomplished with a daisy chain technique (see Figure 8). The priority input (BPRN/) of the highest priority master is tied to ground. The priority output (BPRO/) of the highest priority master is then connected to the priority input (BPRN/) of the next lower priority master, and so on. Any master generating a bus request will set its BPRO/ signal high to the next lower priority master. Any master seeing a high signal on its BPRN/ line will sets its BPRO/ line high, thus passing down priority information to lower priority masters. In this implementation, the bus request line (BREQ/) is not used outside of the individual masters. A limited number of masters can be accommodated by this technique, due to gate delays through the daisy chain. Using the current Intel MULTIBUS controller chip on the master boards up to 3 masters may be accommodated if a BCLK/ period of 100 ns is used. If more bus masters are required, either BCLK/ must be slowed or a parallel priority technique used.

Parallel Priority Technique

In the parallel priority technique, the priority is resolved in a priority resolution circuit in which the highest priority BREQ/ input is encoded with a priority encoder chip (74148). This coded value is then decoded with a priority decoder chip (74S138) to activate the appropriate BPRN/ line. The BPRO/ lines are not used in the parallel priority scheme. However, since the MULTIBUS backplane contains a trace from the BPRN/ signal of one card slot to the BPRO/ signal of the adjacent lower card slot, the BPRO/ must be disconnected from the bus on the board or the backplane trace must be cut. A practical limit of sixteen masters can be accommodated using the parallel priority technique due to physical bus length limitations. Figure 9 contains the schematic for a typical parallel resolution network. Note that the parallel priority resolution network must be externally supplied.

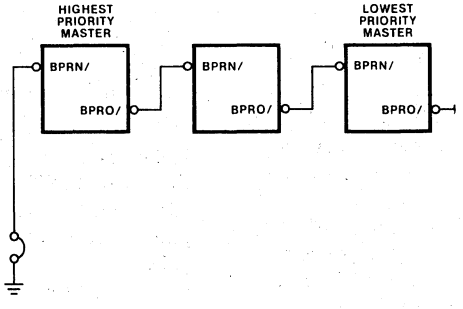


Figure 8. Serial Priority Technique

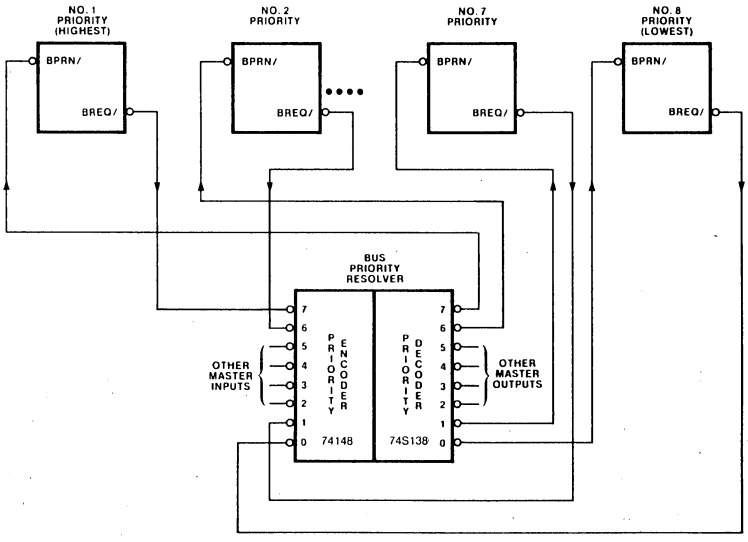


Figure 9. Parallel Priority Technique

MULTIBUS® Exchange Operation — A timing diagram for the MULTIBUS exchange operation is shown in Figure 10. This implementation example uses a parallel resolution scheme, however, the timing would be basically the same for the serial resolution scheme.

In this example, master A has been assigned a lower priority than master B. The bus exchange occurs because master B generates a bus request during a time when master A has control of the bus.

The exchange process begins when master B requires the bus to access some resource such as an I/O or memory module while master A controls the bus. This internal request is synchronized with the trailing edge (high to low) of BCLK/ to generate a bus request (BREQ/). The bus priority resolution circuit changes the BPRN/ signal from active (low) to inactive (high) for master A and from inactive to active for master B. Master A must first complete the current bus command if one is in operation. After master A completes the command, it sets BUSY/ inactive on the next trailing edge of BCLK/. This allows the actual bus exchange to occur, because master A has relinquished control of the bus, and master B has been granted its BPRN/. During this time, the drivers for master A are disabled. Master B must take control of the bus with the next trailing edge of BCLK/ to complete the bus exchange. Master B takes control by activating BUSY/ and enabling its drivers.

It is possible for master A to retain control of the bus and prevent master B from getting control. Master A activates the Bus Override (or Bus Lock) signal which keeps BUSY/ active allowing control of the bus to stay with master A. This guarantees a master consecutive bus cycles for software or hardware functions which require exclusive, continuous access to the bus.

Note that in systems with only a single master it is necessary to ground the BPRN/ pin of the master, if slave boards are to be accessed. In single board systems which use a CPU board capable of Bus Vectored Interrupt operation, the BPRN/ pin must also be grounded.

In a single master system bus transfer efficiency may be gained if the BUS OVERRIDE signal is kept active continuously. This permits the master to maintain control of the bus at all times, therefore saving the overhead of the master reacquiring the bus each time it is needed.

The CBRQ/ line may be used by a master in control of the bus to determine if another master

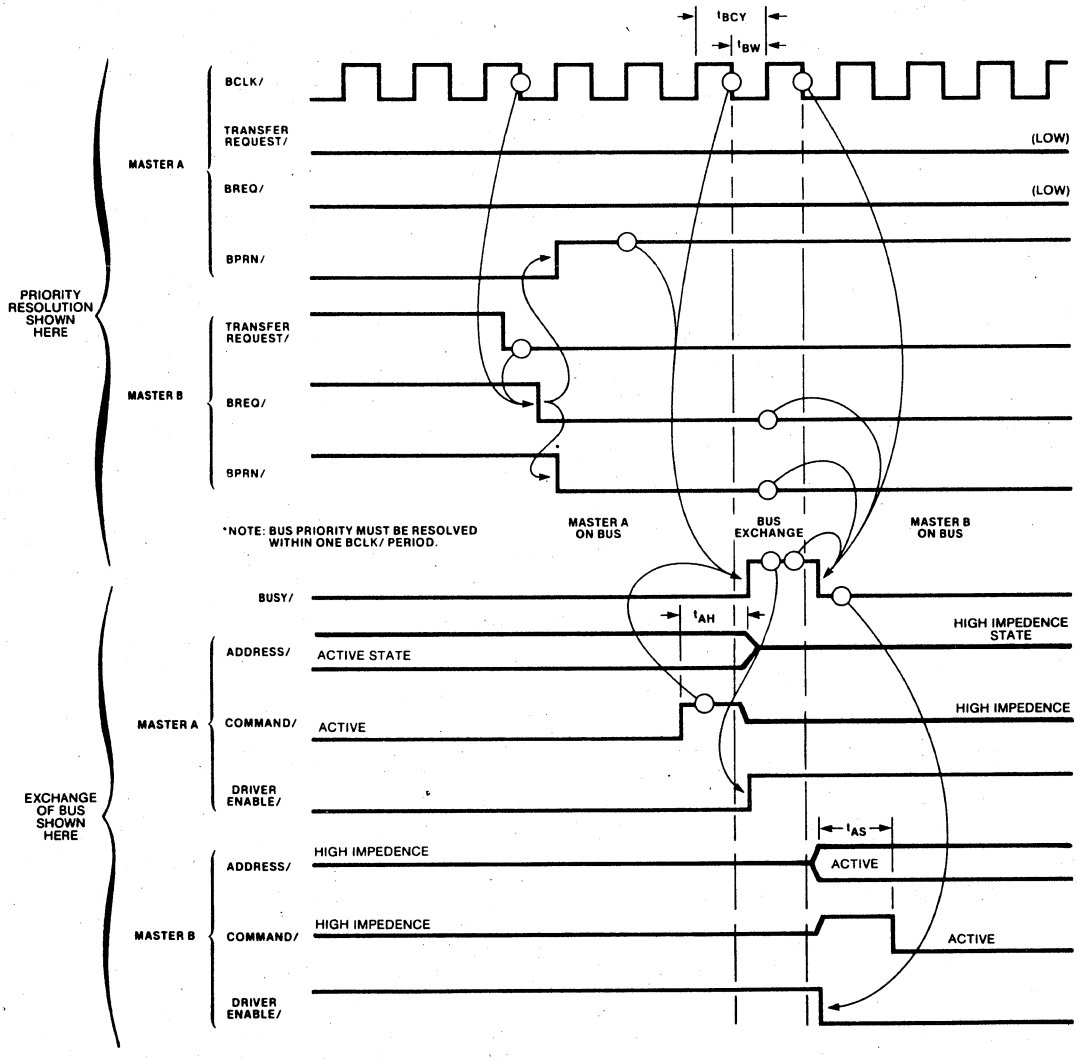


Figure 10. Bus Control Exchange Operation

requires the bus. If a master currently in control of the bus sees the CBRQ/ line inactive, it will maintain control of the bus between adjacent bus accesses. Therefore, when a bus access is required, the master saves the overhead of reacquiring the bus. If a current bus master sees the CBRQ/ line active, it will then relinquish control of the bus after the current bus access and will contend for the bus with the other master(s) requiring the bus. The relative priorities of the masters will determine which master receives the bus.

Note that except for the BUS OVERRIDE state, no single master may keep exclusive control of the bus. This is true because it is impossible for the CPU on a master to require continuous access to the bus. Other lower priority masters will always be able to gain access to the bus between accesses of a higher priority master.

Power Fail Considerations — The MULTIBUS P2 connector signals provide a means of handling power failures. The circuits required for power

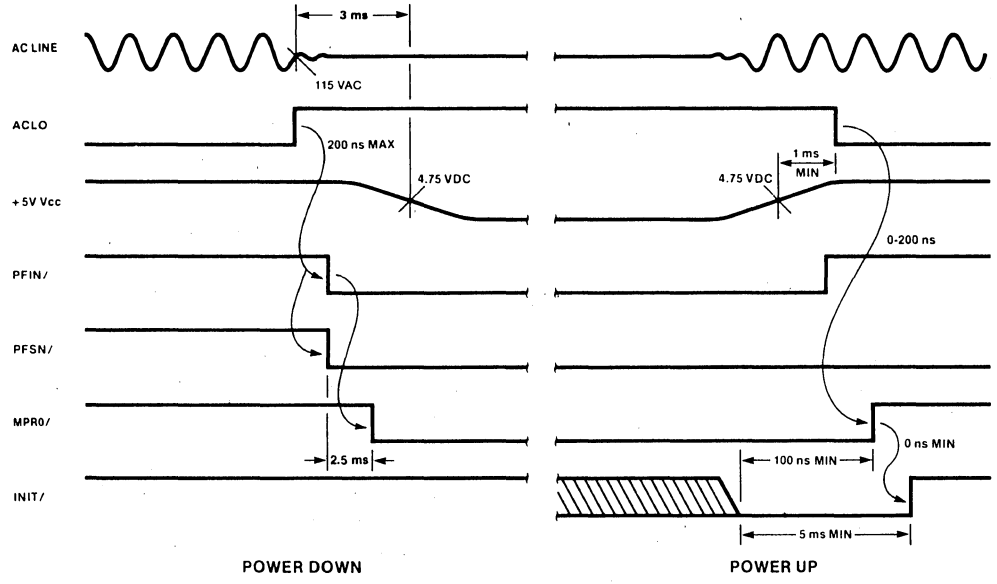


Figure 11. Power Fail Timing Sequence

failure detection and handling are optional and must be supplied by the user. Figure 11 shows the timing of a power fail sequence.

The power supply monitors the AC power level. When power drops below an acceptable value, the power supply raises ACLO which tells the power fail logic that a minimum of three milliseconds will elapse before DC power will fall below regulated voltage levels. The power fail logic sets a sense latch (PFSN/) and generates an interrupt (PFIN/) to the processor so the processor can store its environment. After a 2.5 millisecond timeout, the memory protect signal (MPRO/) is asserted by the power fail logic preventing any memory activity. As power falls, the memory goes on standby power. Note that the power fail logic must be powered from the standby source.

As the AC line revives, the logic voltage level is monitored by the power supply. After power has been at its operating level for one millisecond minimum, the power supply sets the signal ACLO low, beginning the restart sequence. First, the memory protect line (MPRO/) then the initialize line (INIT/) become inactive. The bus master now starts running. The bus master checks the power fail latch (PFSN/) and, if it finds it set, branches to a power up routine which resets the latch (PFSR/), restores the environment, and resumes execution.

Note that INIT/ is activated only after DC power has risen to the regulated voltage levels and must stay low for five milliseconds minimum before the system is allowed to restart. Alternatively, INIT/ may be held low through an open collector device by MPRO/.

How the power failure equipment is configured is left to the system designer. The backup power source may be batteries located on the memory boards or more elaborate facilities located off-board. The location of the power fail logic determines which MULTIBUS power fail lines are used. Pins on the P2 connector have been specified for the power failure functions for use as needed.

To further clarify the location and use of the power fail circuitry, an example of a typical power fail system block diagram is shown in Figure 12. A single board computer and a slave memory board are contained in the system. It is desired to power the memory circuit elements of the memory board from auxiliary power. The single board computer will remain on the main power supply. To accomplish this, user supplied power fail logic and

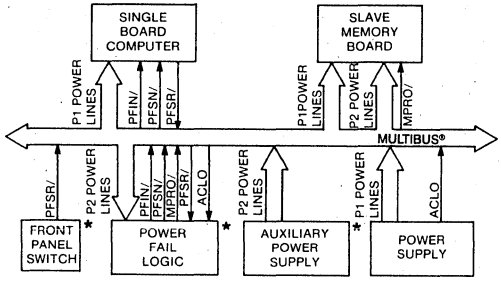


Figure 12. Typical Power Fail System Block Diagram

an auxiliary power supply have been included in the system.

The single board computer is powered from the P1 power lines and accesses the P2 signal lines PFIN/, PFSN/ and PFSR/ (only the P2 signal lines used by a particular functional block are shown on the block diagram). The PFSR/ line is driven from two sources: a front panel switch and the single board computer. The front panel switch is used during normal power-up to reset the power fail sense latch. The single board computer uses the PFSR/ line to reset the latch during a power-up sequence after a power failure. Current single board computers must access the PFSN/ and PFSR/ signals either directly with dedicated circuitry and a P2 pin connection or through the parallel I/O lines with a cable connection from the parallel I/O connector to the P2 connector.

The slave memory board uses both the P1 and P2 power lines, the P2 power lines are used (at all times) to power the memory circuit elements and other support circuits, the P1 power lines power all other circuitry. In addition, the MPRO/ line is input and used to sense when memory contents should be protected.

The power fail logic contains the power fail sense latch, and uses the PFSR/ and ACLO lines for inputs and the PFIN/ PFSN/, and MPRO/ lines for outputs. The power fail logic must be powered by the P2 power lines.

DC Requirements — The drive and load characteristics of the bus signals are listed in Appendix C. The physical locations of the drivers and loads, as well as the terminating resistor value for each bus line, are also specified. Appendix D contains the MULTIBUS power specifications.

MULTIBUS® Slave Interface Circuit Elements

There are three basic elements of a slave bus interface: address decoders, bus drivers, and control signal logic. This section discusses each of these elements in general terms. A description of a detailed implementation of a slave interface is presented in a later section of this application note.

Address Decoding — This logic decodes the appropriate MULTIBUS address bits into RAM requests, ROM requests, or I/O selects. Care must be taken in the design of the address decode logic to ensure flexibility in the selection of base address assignments. Without this flexibility, restrictions may be placed upon various system configurations. Ideally, switches and jumper connections should be associated with the decode logic to permit field modification of base address assignments.

The initial step in designing the address decode portion of a MULTIBUS interface is to determine the required number of unique address locations. This decision is influenced by the fact that address decoding is usually done in two stages. The first stage decodes the base address, producing an enable for the second stage which generates the actual device selects for the user logic. A convenient implementation of this two stage decoding scheme utilizes a pair of decoders driven by the high order bits of the address for the first stage and a second decoder for the low order bits of the address bus. This technique forces the number of unique address locations to be a power of two, based at the address decoded by the first stage. Consider the scheme illustrated in Figure 13.

As shown in Figure 13, the address bits A_4 - A_B are used to produce switch selected outputs of the first stage of decoding. The 1 out of 8 binary decoders

have been used. The top decoder decodes address lines A_4 - A_7, and the bottom decoder decodes address lines A_8 - A_B. If only address lines A_0 - A_7 are being used for device selection, as in the case of I/O port selection in 8-bit systems, the bottom decoder may be disabled by setting switch S2 to the ground position. Address lines A_7 and A_B drive enable inputs E2 or E3 of the decoders. The address lines A_0 - A_3 enter the second stage address decoder to produce 8 user device selects. The second stage decoder must first be enabled by an address that corresponds to the switch-selected base address.

Address decoding must be completed before the arrival of a command. Since the command may become active within 50 ns after stable address, the decode logic should be kept simple with a minimal number of layers of logic. Furthermore, the timing is extremely critical in systems which make use of the inhibit lines.

A linear or unary select scheme in which no binary encoding of device address (e.g., address bit A_0 selects device 0, address bit A_1 selects device 1, etc.) is performed is not recommended because the scheme offers no protection in case multiple devices are simultaneously selected, and because the addressing within such a system is restricted by the extent of the address space occupied by such a scheme.

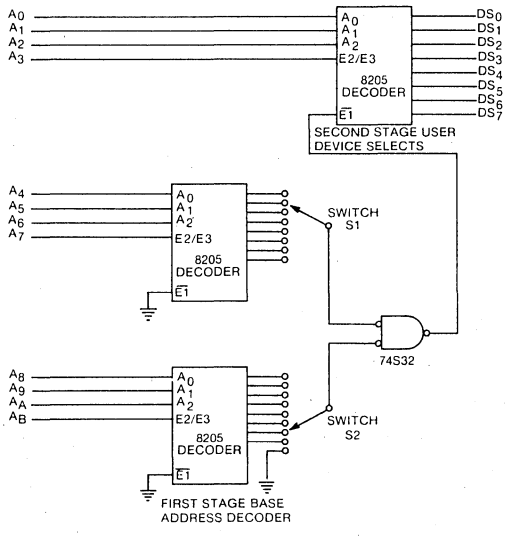


Figure 13. Two Stage Decoding Scheme

Data Bus Drivers — For user designed logic which simply receives data from the MULTIBUS data lines, this portion of the bus interface logic may only consist of buffers. Buffers are required to ensure that maximum allowable bus loading is not exceeded by the user logic.

In systems where the user designed logic must place data onto the MULTIBUS data lines, three-state drivers are required. These drivers should be enabled only when a memory read command (MRDC/) or an I/O read command (IORC/) is present and the module has been addressed.

When both the read and write functions are required, parallel bidirectional bus drivers (e.g., Intel 8226, 8287, etc.) are used. A note of caution must be included for the designer who uses this type of device. A problem may arise if data hold time requirements must be satisfied for user logic following write operations. When bus commands are used to directly produce both the chip select for the bidirectional bus driver and a strobe to a latch in the user logic, removal of that signal may not provide the user's latch with adequate data hold time. Depending on the specifics of the user logic, this problem may be solved by permanently enabling the data buffer's receiver circuits and controlling only the direction of the buffers.

Control Signal Logic — The control signal logic consists of the circuits that forward the I/O and memory read/write commands to their respective destinations, provide the bus with a transfer acknowledge response, and drive the system interrupt lines.

Bus Command Lines

The MULTIBUS information transfer protocol lines (MRDC/, MWTC/, IORD/. and IOWC/) should be buffered by devices with very high speed switching. Because the bus DC requirements specify that each board may load these lines with 2.0 mA, Schottky devices are recommended. LS devices are not recommended due to their poor noise immunity. The commands should be gated

with a signal indicating the base address has been decoded to generate read and write strobes for the user logic.

Transfer Acknowledge Generation

The user interface transfer acknowledge generation logic provides a transfer acknowledge response, XACK/, to notify the bus master that write data provided by the bus master has been accepted or that read data it has requested is available on the MULTIBUS data lines. XACK/ allows the bus master to conclude its current instruction.

Since XACK/ timing requirements depend on both the CPU of the bus master and characteristics of the user logic, a circuit is needed which will provide a range of easily modified acknowledge responses.

The transfer acknowledge signals must be driven by three-state drivers which are enabled when the bus interface is addressed and a command is present.

Interrupt Signal Lines

The asynchronous interrupt lines must be driven by open collector devices with a minimum drive of 16 mA.

In a typical Non Bus Vectored Interrupt system, logic must be provided to assert and latch-up an interrupt signal. In addition to driving the MULTIBUS interrupt lines, the latched interrupt signal would be read by an I/O operation such as reading the module's status. The interrupt signal would be cleared by writing to the status register.

III. MULTIBUS® SLAVE DESIGN EXAMPLE

A MULTIBUS slave design example has been included in this application note to reinforce the theory previously discussed. The design example is of general purpose I/O slave interface. This design example could easily be modified to be used as a slave memory interface by buffering the address signals and using the appropriate MULTIBUS memory commands. In addition, to help the reader better understand an application for an I/O slave interface, two Intel 8255A Parallel Peripheral Interface (PPI) devices are shown connected to the slave interface.

The design example is shown in both 8/16-bit version and an 8-bit version. The 8/16-bit version is an I/O interface which will permit a 16-bit master to perform 8 or 16 bit data transfers. 8-bit masters may also use the 8/16-bit version of the design example to perform 8-bit data transfers.

The 8-bit version of the design example may be used by both 8 or 16-bit masters, but will only perform 8-bit data transfers. It does not contain the circuitry required to perform 16-bit data transfers.

Both the 8/16-bit version and the 8-bit version of the design example were implemented on an iSBC 905 prototype board. The schematics for each of the examples are given in Appendices F and G.

Functional/Programming Characteristics

This section describes the organization of the slave interface from two points of view, the functional point of view and the programming characteristics. First, the principal functions performed by the hardware are identified and the general data flow is illustrated. This point of view is intended as an introduction to the detailed description provided in the next section; Theory of Operation. In the second point of view, the information needed by a programmer to access the slave is summarized.

Functional Description — The function of this I/O slave is to provide the bus interface logic for general purpose I/O functions and for two Intel 8255A Parallel Peripheral Interface (PPI) devices. Eight device selects (port addresses) are available for general purpose I/O functions. One of these device select lines is used to read and reset the state of an interrupt status flip-flop, the other seven device selects are unused in this design. An additional eight I/O device port addresses are used by the two 8255A devices; four I/O port addresses per 8255A (three I/O port address for the three parallel ports A, B, and C and the fourth I/O port address for the device control register).

Figure 14 contains a functional block diagram of the slave design example. This block diagram shows the fundamental circuit elements of a bus slave: bidirectional data bus drivers/receivers, address decoding logic and bus control logic. Also shown is the address decoding logic for the low order four bits, the interrupt logic which is selected by this decoding logic, and the two 8255A devices.

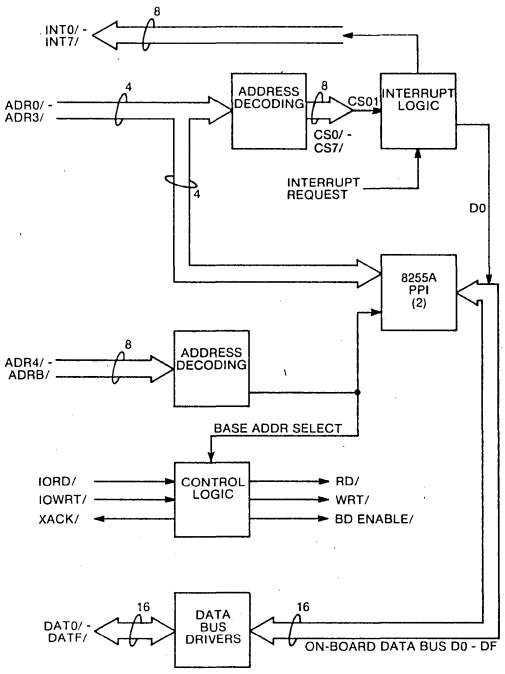


Figure 14. MULTIBUS® Slave Design Example Functional Block Diagram

Programming Characteristics — The slave design example provides 16 I/O port addresses which may be accessed by user software. The base address of the 16 contiguous port addresses is selected by wire wrap connections on the prototype board. The wire wrap connections specify address bits ADR4/ - ADRB/. They allow the selection of a base address on any 16 byte boundary. Twelve address bits (ADR0/ - ADRB/) are used since 16-bit (8086 based) masters use 12 bits to specity I/O port addresses. If an 8 bit (8080 or 8085 based) master is used with this slave board, the high order address bits (ADR8/ - ADRB/) must not be used by the decoding circuits; a wire wrap jumper position (ground position) is provided for this.

The 16 I/O port addresses are divided into two groups of 8 port addresses by decoding address line ADR3/. Port addresses XX0 - XX7 are used for general I/O functions (XX indicates any hexidecimal digit combination). Port address XX0 is used for accessing the interrupt status flip-flop and port addresses XX1 - XX7 are not used in this example. Port addresses XX8 - XXF are used for accessing the PPIs. If port addresses XX8 - XXF are selected, then ADR0/ is used to specify which of two PPIs are selected. If the address is even (XX8, XXA, XXC, or XXE) then one PPI is selected. If the address is odd (XX9, XXB, XXD, or XXF), then the other PPI is selected. ADR1/ and ADR2/ are connected directly to the PPIs. Table 1 summarizes the I/O port addresses of the slave design example. Note that if a 16-bit master is used, it is possible to access the slave in a byte or word mode. If word access is used with port address XX8, XXA, XXC, or XXE, then 16 bit transfers will occur between the PPIs and the master. These 16 bit transfers occur because an even address has been specified and the MULTIBUS BHEN/ signal indicates that a 16-bit transfer is requested.

Theory of Operation

In the preceding section, each of the slave design example functional blocks was identified and briefly explained. This section explains how these functions are implemented. For detailed circuit information, refer to the schematics in Appendices F and G. The schematic in Appendix F is on a foldout page so that the following text may easily be related to the schematic.

The discussion of the theory of operation is divided into five segments, each of which discusses a different function performed by the MULTIBUS slave design example. The five segments are:

1. Bus address decoding
2. Data buffers
3. Control signals
4. Interrupt logic
5. PPI operation

Each of these topics are discussed with regard to the 8/16-bit version of the design example; followed by a discussion of the circuit elements which are required by the 8-bit version of the interface.

Bus Address Decoding — Bus address decoding is performed by two 8205 1 out of 8 binary decoders. One decoder (A3) decodes address bits ADR8/ - ADRB/ and the second decoder (A2) decodes address bits ADR4/ - ADR7/. The base address

Table 1
SLAVE DESIGN EXAMPLE PORT ADDRESSES

I/O PORT ADDRESS	READ	WRITE
BYTE ACCESS		
XX0	Bit 0 = Interrupt Status	Reset Interrupt Status
XX1 - XX7	Unused	Unused
XX8	Parallel Port A, Even PPI	Parallel Port A, Even PPI
XX9	Parallel Port A, Odd PPI	Parallel Port A, Odd PPI
XXA	Parallel Port B, Even PPI	Parallel Port B, Even PPI
XXB	Parallel Port B, Odd PPI	Parallel Port B, Odd PPI
XXC	Parallel Port C, Even PPI	Parallel Port C, Even PPI
XXD	Parallel Port C, Odd PPI	Parallel Port C, Odd PPI
XXE	Illegal Condition	Control, Even PPI
XXF	Illegal Condition	Control, Odd PPI
WORD ACCESS		
XX0	Bit 0 = Interrupt Status	Reset Interrupt Status
XX2 - XX6	Unused	Unused
XX8	Parallel Port A, Even and Odd PPIs	Parallel Port A, Even and Odd PPIs
XXA	Parallel Port B, Even and Odd PPIs	Parallel Port B, Even and Odd PPIs
XXC	Parallel Port C, Even and Odd PPIs	Parallel Port C, Even and Odd PPIs
XXE	Illegal Condition	Control, Even and Odd PPIs

XX = Any hex digits, assigned by jumpers; XX defines the base address.

selected is determined by the position of wire wrap jumpers. The outputs of the two decoders are ANDed together to form the BASE ADR SELECT/ signal. This signal specifies the base address for a group of 16 I/O ports. Using the wire wrap jumper positions shown in the schematic, a base address of E3 has been selected. Therefore, this MULTIBUS slave board will respond to I/O port addresses in the E30 - E3F range.

If this slave board is to be used with 8-bit MULTIBUS masters, the high order address bits must not be decoded. Therefore, the wire wrap jumper which selects the output of decoder A3 must be placed in the top (ground) position (pin 10 of gate A9 to ground).

The low order 4 address lines (ADR0/ - ADR3/) are buffered and inverted using 74LS04 inverters. These address lines are input to an 8205 for decoding a chip select for the interrupt logic; the address lines are also used directly by the PPIs. LS-Series logic is required for buffering to meet the MULTIBUS specification for I_{IL} (low level input current). S-Series or standard series logic will not meet this specification.

Address decoder A4 is used to decode addresses E30 - E37. The CS0/ output of this decoder is used to select the interrupt logic, thus I/O port address E30 is used to read and reset the interrupt latch. The remaining outputs from decoder A4 (CS1/ - CS7/) are not used in this example. They would normally be used to select other functions in a slave board with more capability. Note that in the schematic shown in Appendix G for the 8-bit version of this slave design example, the high order (ADR8/ - ADRB/) address decoder is not included and the BHEN/ signal is not used.

Data Buffers — Intel 8287 8-bit parallel bi-directional bus drivers are used for the MULTIBUS data lines DAT0/ - DATF/. In the 8/16-bit version of the slave board, three 8287 drivers are used.

When an 8-bit data transfer is requested, either driver A5, which is connected to on-board data

lines D0 - D7, or driver A6, which is connected to on-board data lines D8 - DF, is used. If a byte transfer is requested from an even address, driver A5 will be selected. If a byte transfer from an odd address is requested, driver A6 will be selected. All byte transfers take place on MULTIBUS data lines DAT0/ - DAT7/. When a word (16-bit) transfer is requested from an even address, drivers A5 and A7 will be used. Note that if a user program requests a word transfer from an odd address, 16-bit masters in the iSBC product line will actually perform two byte transfer requests.

The logic which determines the chip selection (8287 input signal OE, output enable) signals for the bus drivers uses the low order address bit (ADR0/) and the buffered Byte High Enable signal (BHENBL/). Note that the MULTIBUS signal BHEN/ has been buffered with an 74LS04 inverter. This is done to meet the bus address line loading specification. The SWAP BYTE/ signal which is generated is qualified by the BD ENBL/ signal and used to select the bus drivers.

The steering pin for the 8287 drivers is labelled T (transmit) and is driven by the signal RD. When an input (read) request is active or when neither a read or write command is being serviced, the direction of data transfer of the 8287 will be set for B to A.

The 8287 drivers are set to point IN (direction B to A) when no MULTIBUS I/O transfer command is being serviced for two reasons. First, if the driver were pointed OUT (direction A to B) and a write command occured, it would be necessary to turn the buffers IN and set the OE (output enable) signal active before the data could be transferred to the on-board bus. A possibility of a "buffer-fight" could occur in some designs if the OE signal permitted an 8287 to drive the MULTIBUS data lines momentarily before the steering signal could switch the direction of the 8287. In this case, both the MULTIBUS master and the slave would be driving the data lines; this is not recommended. (In this particular design, the steering signal will always stabilize before the OE signal becomes active.)

The second reason the driver is pointing IN when no command is present is due to the "data valid after WRITE" requirements of the 8255As. The 8255A requires that data remain on its data lines for 30 ns after the WRITE command ($\overline{WR}$ at the 8255A) is removed. This requirement will be met if the direction of the 8287 drivers is not switched when the MULTIBUS IOWC/ signal is removed (WRT/ could have been used to steer the 8287 instead of RD); and if the capacitance of the on-board data bus lines is sufficient to hold the data values on the bus after the 8287 OE signal and the 8255A PPI WRT/ signal go inactive. The on-board data bus may easily be designed such that the capacitance of the lines is sufficient to meet the 30 ns data hold time requirement. In addition, the current leakage of all devices connected to the on-board bus must be kept small to meet the 30 ns data hold time requirement.

The 8-bit version of this design example uses only one 8287 instead of the three required by the 8/16-bit version. The logic required to control the swap byte buffer is also not necessary. The chip select signal used for the 8287 is the BD ENBL/ signal.

Control Signals — The MULTIBUS control signals used by this slave design example are IORC/, IOWC/, and XACK/. IORC/ and IOWC/ are qualified by the BASE ADR SELECT/ signal to form the signals RD and WRT. RD and WRT are used to drive the interrupt logic, the PPI logic and the XACK/ (transfer acknowledge) logic.

For the XACK/ logic RD and WRT are ORed to form the BD ENBL/ signal which is inverted and used to drive the CLEAR pin of a shift register. When the slave board is not being accessed, the CLEAR pin of the shift register will be low (BD ENBL/ is high). This causes the shift register to remain cleared and all outputs of the shift register will be low. When the slave board is accessed, the CLEAR pin will be high, and the A and B inputs (which are high) will be clocked to the output pins by CCLK/. To select a delay for the XACK/ signal, a jumper must be installed from one of the shift register output pins to the 8089 tri-state driver. Each of the shift register output pins select an integer multiple of CCLK/ periods for the signal delay. Since the CCLK/ signal is asynchronous, the actual delay selected may only be specified with a tolerance of one CCLK/ period. In this example a delay of 3 - 4 CCLK/ periods was selected; with a CCLK/ period of 100 ns, the XACK/ delay would occur somewhere within the range of 300 - 400 ns from the time when the CLEAR signal goes high.

The control signal logic used in the 8-bit version of the slave design example is identical to the logic used in the 8/16-bit version.

Interrupt Logic — The interrupt logic uses a 74S74 flip-flop to latch an asynchronous interrupt request from some external logic. The Q output of the INTERRUPT REQUEST LATCH is output through an open collector gate to one of the MULTIBUS interrupt lines. The state of the INTERRUPT REQUEST LATCH is transferred to the INTERRUPT STATUS LATCH when a read command is performed on I/O port BASE ADDRESS+0 (E30 for the jumper configuration shown). The Q output of INTERRUPT STATUS LATCH is used to drive data line D0 of the on-board data bus by using an 8089 tri-state driver. If a user program performs an INPUT from I/O port E30, data bit 0 will be set to 1 if the INTERRUPT REQUEST LATCH is set.

The purpose of INTERRUPT STATUS LATCH is to minimize the possibility of the asynchronous interrupt occuring while the interrupt status is being read by a bus master. If the latch was not included in the design and an asynchronous interrupt did occur while a bus master is reading MULTIBUS data line DAT0/, a data buffer on the master could go into a meta-stable state. By adding the extra latch, which is clocked by the IORD/ command for I/O port E30, the possibility of data line DAT0/ changing during a bus master read operation is eliminated.

The INTERRUPT REQUEST LATCH is cleared when a user program performs an OUTPUT to I/O port E30.

This interrupt structure assumes that several interrupt sources may exist on the same MULTIBUS interrupt line (for example, INT3/). When the MULTIBUS master gets interrupted, it must poll the possible sources of the interrupt received and after determining the source of the interrupt, it must clear the INTERRUPT REQUEST LATCH for that particular interrupt source.

The interrupt logic for the 8-bit version of the design example is identical to the interrupt logic of the 8/16-bit version of the design example.

PPI Operation — Two 8255A Parallel Peripheral Interface (PPI) devices are shown interfaced to the slave design example logic. One PPI is connected to the on-board data bus lines D0 - D7 and is addressed with the even I/O port addresses E38, E3A, E3C, and E3E. The second PPI is connected to data bus lines D8 - DF and is addressed with the odd I/O port addresses E39, E3B, E3D, and E3F. The even or odd I/O port selection is controlled by using the ADR0 address line in the chip select term of the PPIs. In addition, the odd PPI (A11) is selected when the BHENBL term is high. This occurs when the MULTIBUS signal BHEN/ is low indicating that a word (16-bit) I/O instruction is being executed. When a word I/O instruction is executed, both PPIs will perform the I/O operation specified.

The specifications of the 8255A device state that the address lines A0 and A1 and the chip select lines must be stable before the $\overline{RD}$ or $\overline{WR}$ lines are activated. The MULTIBUS specification address set-up time of 50 ns and the short gate propagation delays in this design assure that the address lines are stable before $\overline{RD}$ or $\overline{WR}$ are active.

The data hold requirements of the 8255A were discussed in a previous section. The 8255A specification states that data will be stable on the data bus lines a maximum of 250 ns after a READ command. This specification was used to select the delay for the XACK/ signal.

The PPI operation for the 8-bit version of the design example is slightly different than that used for the 8/16-bit version. The chip select signal for the bottom PPI does not use the BHENBL term since 16-bit data transfers are not possible with an 8-bit I/O slave board. Also, the chip select and address signals have been swapped so the top PPI occupies I/O address range X8 - XB, and the bottom PPI occupies I/O address range XC - XF (X is the base address of the 8-bit version). This swapping of the address lines was not necessary; however, it was thought to be more convenient to access the PPIs in two groups of 4 contiguous I/O port addresses.

IV. SUMMARY

This application note has shown the structure of the Intel MULTIBUS system bus. The structure supports a wide range of system modules from the Intel OEM Microcomputer Systems product line that can be extended with the addition of user designed modules. Because the user designed modules are no doubt unique to particular applications, a goal of this application note has been to describe in detail the singular common element - the bus interface. Material has also been presented to assist the systems designer to understanding the bus functions so that successful systems integration can be achieved.

Appendix

Contents

APPENDIX A

PIN ASSIGNMENT OF BUS SIGNALS ON MULTIBUS® BOARD P1 CONNECTOR

	(COMPONENT SIDE)			(CIRCUIT SIDE)		
	PIN	MNEMONIC	DESCRIPTION	PIN	MNEMONIC	DESCRIPTION
POWER SUPPLIES	1	GND	Signal GND	2	GND	Signal GND
	3	+5V	+5Vdc	4	+5V	+5Vdc
	5	+5V	+5Vdc	6	+5V	+5Vdc
	7	+12V	+12Vdc	8	+12V	+12Vdc
	9	−5V	−5Vdc	10	−5V	−5Vdc
	11	GND	Signal GND	12	GND	Signal GND
BUS CONTROLS	13	BCLK/	Bus Clock	14	INIT/	Initialize
	15	BPRN/	Bus Pri. In	16	BPRO/	Bus Pri. Out
	17	BUSY/	Bus Busy	18	BREQ/	Bus Request
	19	MRDC/	Mem Read Cmd	20	MWTC/	Mem Write Cmd
	21	IORC/	I/O Read Cmd	22	IOWC/	I/O Write Cmd
	23	XACK/	XFER Acknowledge	24	INH1/	Inhibit 1 disable RAM
BUS CONTROLS AND ADDRESS	25		Reserved	26	INH2/	Inhibit 2 disable PROM or ROM
	27	BHEN/	Byte High Enable	28	AD10/	
	29	CBRQ/	Common Bus Request	30	AD11/	Address
	31	CCLK/	Constant Clk	32	AD12/	Bus
	33	INTA/	Intr Acknowledge	34	AD13/	
INTERRUPTS	35	INT6/	Parallel	36	INT7/	Parallel
	37	INT4/	Interrupt	38	INT5/	Interrupt
	39	INT2/	Requests	40	INT3/	Requests
	41	INT0/		42	INT1/	
ADDRESS	43	ADRE/		44	ADRF/	
	45	ADRC/		46	ADRD/	
	47	ADRA/	Address	48	ADRB/	Address
	49	ADR8/	Bus	50	ADR9/	Bus
	51	ADR6/		52	ADR7/	
	53	ADR4/		54	ADR5/	
	55	ADR2/		56	ADR3/	
	57	ADR0/		58	ADR1/	
DATA	59	DATE/		60	DATF/	
	61	DATC/		62	DATD/	
	63	DATA/	Data	64	DATB/	Data
	65	DAT8/	Bus	66	DAT9/	Bus
	67	DAT6/		68	DAT7/	
	69	DAT4/		70	DAT5/	
	71	DAT2/		72	DAT3/	
	73	DAT0/		74	DAT1/	
POWER SUPPLIES	75	GND	Signal GND	76	GND	Signal GND
	77		Reserved	78		Reserved
	79	−12V	−12Vdc	80	−12V	−12Vdc
	81	+5V	+5Vdc	82	+5V	+5Vdc
	83	+5V	+5Vdc	84	+5V	+5Vdc
	85	GND	Signal GND	86	GND	Signal GND

APPENDIX A (Continued)

P2 CONNECTOR PIN ASSIGNMENT OF OPTIONAL BUS SIGNALS

	(COMPONENT SIDE)			(CIRCUIT SIDE)	
PIN	MNEMONIC	DESCRIPTION	PIN	MNEMONIC	DESCRIPTION
1	GND	Signal GND	2	GND	Signal GND
3	5 VB	+5V Battery	4	5 VB	+5V Battery
5		Reserved	6	VCCPP	+5V Pulsed Power
7	-5 VB	-5V Battery	8	-5 VB	-5V Battery
9		Reserved	10	Reserved	
11	12 VB	+12V Battery	12	12 VB	+12V Battery
13	PFSR/	Power Fail Sense Reset	14	Reserved	
15	-12 VB	-12V Battery	16	-12 VB	-12V Battery
17	PFSN/	Power Fail Sense	18	ACLO	AC Low
19	PFIN/	Power Fail Interrupt	20	MPRO/	Memory Protect
21	GND	Signal GND	22	GND	Signal GND
23	+15V	+15V	24	+15V	+15V
25	-15V	-15V	26	-15V	-15V
27	PAR1/	Parity 1	28	HALT/	Bus Master HALT
29	PAR2/	Parity 2	30	WAIT/	Bus Master WAIT STATE
31			32	ALE	Bus Master ALE
33			34	Reserved	
35			36	Reserved	
37			38	AUX RESET/	Reset switch
39			40		
40			42		
43	Reserved		44		
45			46		
47			48		
49			50	Reserved	
51			52		
53			54		
55			56		
57			58		
59			60		

Notes:

1. PFIN, on slave modules, if possible, should have the option of connecting to INT0/ on P1.
2. All undefined pins are reserved for future use.

APPENDIX B
BUS TIMING SPECIFICATIONS SUMMARY

Parameter	Description	Minimum	Maximum	Units
t_{BCY}	Bus Clock Period	100	D.C.	ns
t_{BW}	Bus Clock Width	0.35 t_{BCY}	0.65 t_{BCY} (Not Restricted)	
t_{SKEW}	BCLK/skew		3	ns
t_{PD}	Standard Bus Propagation Delay		3	
t_{AS}	Address Set-Up Time (at Slave Board)	50		ns
t_{DS}	Write Data Set Up Time	50		ns
t_{AH}	Address Hold Time	50		ns
t_{DHW}	Write Data Hold Time	50		ns
t_{DXL}	Read Data Set Up Time To XACK	0		ns
t_{DHR}	Read Data Hold Time	0	65	ns
t_{XAH}	Acknowledge Hold Time	0	65	ns
t_{XACK}	Acknowledge Time	0	8	μs
t_{CMD}	Command Pulse Width	100	9.5	ns
t_{ID}	Inhibit Delay	0	100 (Recommend < 100 ns)	ns
t_{XACKA}	Acknowledge Time of of an Inhibited Slave	t_{IAD} + 50 ns	1500	
t_{XACKB}	Acknowledge Time of an Inhibiting Slave	1.5	8	μs
t_{IAD}	Acknowledge Disable from Inhibit (An internal parameter on an inhibited slave; used to determine t_{XACKA} Min.)	0	100 (arbitrary)	ns
t_{AIZ}	Address to Inhibits High Delay		100	ns
t_{INTA}	INTA/ Width	250		ns
t_{CSEP}	Command Separation	100		ns

APPENDIX B (Continued)

BUS TIMING SPECIFICATIONS SUMMARY

Parameter	Description	Minimum	Maximum	Units
t_{BREQL}	↓BCLK/ to BREQ/ Low Delay	0	35	ns
t_{BREQH}	↓BCLK/ to BREQ/ High Delay	0	35	ns
t_{BPRNS}	BPRN/ to ↓BCLK/ Setup Time	22		ns
t_{BUSY}	BUSY/ delay from ↓BCLK/	0	70	ns
t_{BUSYS}	BUSY/ to ↓BCLK/ Setup Time	25		ns
t_{BPRO}	↓BCLK/ to BPRO/ (CLK to Priority Out)	0	40	ns
t_{BPRNO}	BPRN/ to BPRO/ (Priority In to Out)	0	30	ns
t_{CBRO}	↓BCLK/ to CBRQ/ (CLKto Common Bus Request)	0	60	ns
t_{CBRQS}	CBRQ/ to ↓BCLK/ Setup Time	35		ns
t_{XCD}	XACK↓ to Command↑ Delay	0	1500	ns
t_{BSYO}	CBRQ/↓ and BUSY/↓ to BUSY/↑	—	12	μs
t_{CCY}	C-clock Period	100	110	ns
t_{CW}	C-clock Width	0.35 t_{CCY}	0.65 t_{CCY}	ns
t_{INIT}	INIT/Width	5		ms
t_{INITS}	INIT/ to MPRO/ Setup Time	100		ns
t_{PBD}	Power Backup Logic Delay	0	200	ns
t_{PFINW}	PFIN/ Width	2.5		ms
t_{MPRO}	MPRO/ Delay	2.0	2.5	ms
t_{ACLOW}	ACLO/ Width	3.0		ms
t_{PFSRW}	PFSR/ Width	100		ns
t_{TOUT}	Timeout Delay	5	∞ (D.C.)	ms
t_{DCH}	D.C. Power Supply Hold from ALCO/	3.0		ms
t_{DCS}	D.C. Power Supply Setup to ACLO/	5		ms

APPENDIX C

BUS DRIVERS, RECEIVERS, AND TERMINATIONS

Driver 1,3						Receiver 2,3				Termination			
Bus Signals	Location	Type	I_{OL} Min_{ma}	I_{OH} $Min_{\mu a}$	C_O Max_{pf}	Location	I_{IL} Max_{ma}	I_{IH} $Max_{\mu a}$	C_I Max_{pf}	Location	Type	R	Units
DAT0/→DATF/ (16 lines)	Masters and Slaves	TRI	16	-2000	300	Masters and Slaves	-0.8	125	18	1 place	Pullup	2.2	KΩ
ADR0/-ADRB/	Masters	TRI	16	-2000	300	Slaves	-0.8	125	18	1 place	Pullup	2.2	KΩ
BHEN/ (21 lines)													
MRDC/,MWTC/	Masters	TRI	32	-2000	300	Slaves (Memory; memory-mapped I/O)	-2	125	18	1 place	Pullup	1	KΩ
IORC/,IOWC/	Masters	TRI	32	-2000	300	Slaves (I/O)	-2	125	18	1 place	Pullup	1	KΩ
XACK/	Slaves	TRI	32	-2000	300	Masters	-2	125	18	1 place	Pullup	510	Ω
INH1/,INH2/	Inhibiting Slaves	OC	16	—	300	Inhibited Slaves (RAM, PROM, ROM, Memory-Mapped I/O)	-2	50	18	1 place	Pullup	1	KΩ
BCLK/	1 place (Master us)	TTL	48	-3000	300	Master	-2	125	18	Mother-board	To +5V To GND	220 330	Ω Ω
BREQ/	Each Master	TTL	5	-200	60	Central Priority Module	2	50	18	Central Priority Module (not req)	Pullup	1	KΩ
BPRO/	Each Master	TTL	5	-200	60	Next Master in Serial Priority Chain at its BPRN/	-1.6	50	18	(not req)			
BPRN/	Parallel: Central Priority Module Serial:Prev Masters BPRO/	TTL	5	-200	300	Master	-4	100		(not req)			
BUSY/, CBRQ	All Masters	O.C.	20	—	300	All Masters	-2	50	18	1 place	Pullup	1	KΩ
INIT/.	Master.	O.C.	32	—	300	All	-2	50	18	1 place	Pullup	2.2	KΩ
CCLK/	1 place	TTL	48	-3000	300	Any	-2	125	18	Mother-board	To +5V To GND	220 330	Ω Ω
INTA/	Masters	TRI	32	-2000	300	Slaves (Interrupting I/O)	-2	125	18	1 place	Pullup	1	KΩ
INT0/→INT7/ (8 lines)	Slaves	O.C.	16	—	300	Masters	-1.6	40	18	1 place	Pullup	1	KΩ
PFSR/	User's Fron Panel?	TTL	16	-400	300	Slaves, Masters	-1.6	40	18	1 place	Pullup	1	KΩ
PFSN/	Power Back Up Unit	TTL	16	-400	300	Masters	-1.6	40	18	1 place	Pullup	1	KΩ
ACLO	Power Supply	O.C.	16	-400	300	Slaves, Masters	-1.6	40	18	1 place	Pullup	1	KΩ
PFIN/	Power Back-Up Unit	O.C.	16	-400	300	Masters	-1.6	40	18	1 place	Pullup	1	KΩ
MPRO/	Power Back-Up Unit	TTL	16	-400	300	Slaves Masters	-1.6	40	18	1 place	Pullup	1	KΩ

APPENDIX C (Continued)
BUS DRIVERS, RECEIVERS, AND TERMINATIONS

Driver 1,3						Receiver 2,3				Termination		
Bus Signals	Location	Type	I_{OL} Min_{ma}	I_{OH} $Min_{\mu a}$	C_O Max_{pf}	Location	I_{IL} Max_{ma}	I_{IH} $Max_{\mu a}$	C_I Max_{pf}	Location	Type	R Units
Aux Reset/	User's Front Panel?	Switch to GND (Note 5)	—	—	—	Masters	-2	50	18	None		

Notes:

1. Driver Requirements
   - I_{OH} = High Output Current Drive
   - I_{OL} = Low Output Current Drive
   - C_O = Capacitance Drive Capability
   - TRI = 3-State Drive
   - O.C. = Open Collector Driver
   - TTL = Totem-pole Driver
2. Receiver Requirements
   - I_{IH} = High Input Current Load
   - I_{IL} = Low Input Current Load
   - C_I = Capacitive Load
3. TTL low state must be $\geq$ -0.5v but $\leq$ 0.8v at the receivers
   TTL high state must be $\geq$ 2.0v but $\leq$ 5.5v at the receivers
4. For the iSBC 80/10 and the iSBC 80/10A use only a 1K pull-up resistor to +5v for BCLK/ and CCLK/ termination.
5. Recommend a 47Ω resistor in series with switch.

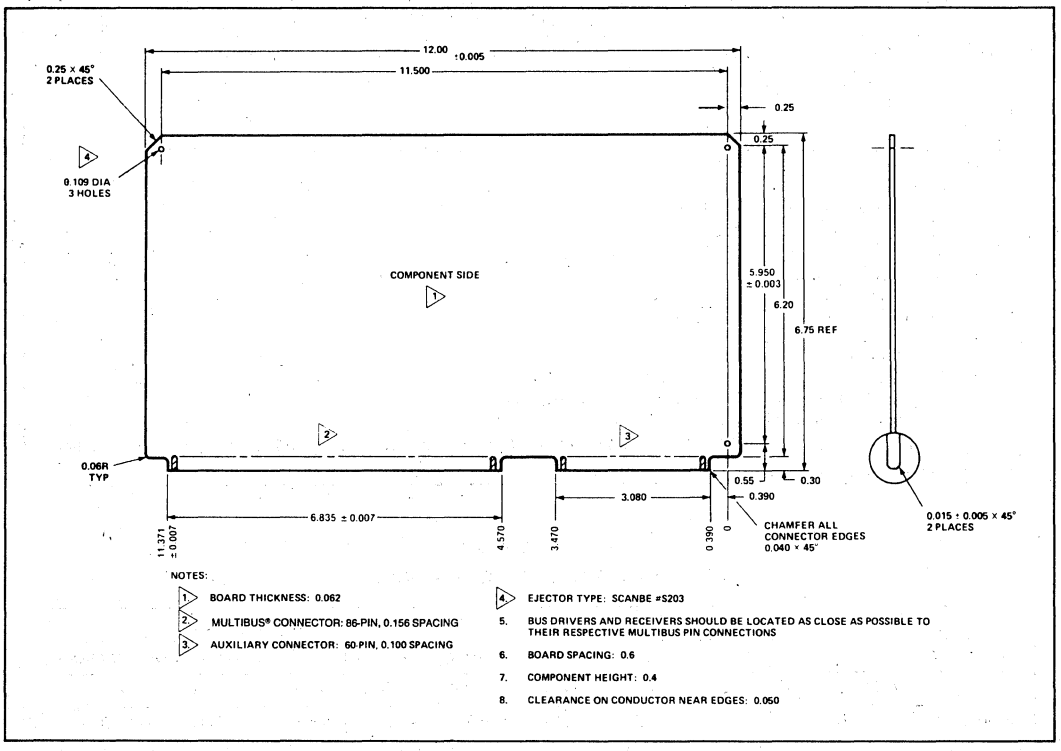
APPENDIX D
BUS POWER SPECIFICATIONS

	Standard (P1)				Optional (P2)					
					Analog Power		Battery Power Backup			
	Ground	+5	+12	−12	+15	−15	+5	+12	−12	−5
Mnemonic	GND	+5V	+12V	−12V	+15V	−15V	+5B	+12B	−12B	−5B
Bus Pins	P1+1,2, 11,12, 75,76 85,86	P1+3,4, 5,6,81, 82,83, 84	P1+7,8	P1+79, 80	P2+23, 24	P2+25, 26	P2+3,4, 5,6	P2+11, 12	P2+15, 16	P2−7,8
Nominal Output	Ref.	+5.0V	+12.0V	−12.0V	+15.0V	−15.0V	+5.0V	+12.0V	−12.0V	−5.0V
Tolerance from Nominal[1]	Ref.	±5%	±5%	±5%	±3%	±3%	±5%	±5%	±5%	±5%
Ripple (Pk-Pk)[2]	Ref.	50 mV	50 mV	50 mV	10 mV	10 mV	50 mV	50 mV	50 mV	50 mV
Transient Response Time[3]		500 μs	500μs	500 μs	100 μs	100 μs	500 μs	500 μs	500 μs	500 μs
Transient Deviation[4]		±10%	±10%	±10%	±10%	±10%	±10%	±10%	±10%	±10%

NOTES:

1. Tolerance is worst case, including initial voltage setting line and load effects of power source, temperature drift, and any additional steady state influences.
2. As measured over any bandwidth not to exceed 0 to 500 kHz.
3. As measured from the start of a load change to the time an output recovers within ±0.1% of final voltage.
4. Measured as the peak deviation from the initial voltage.

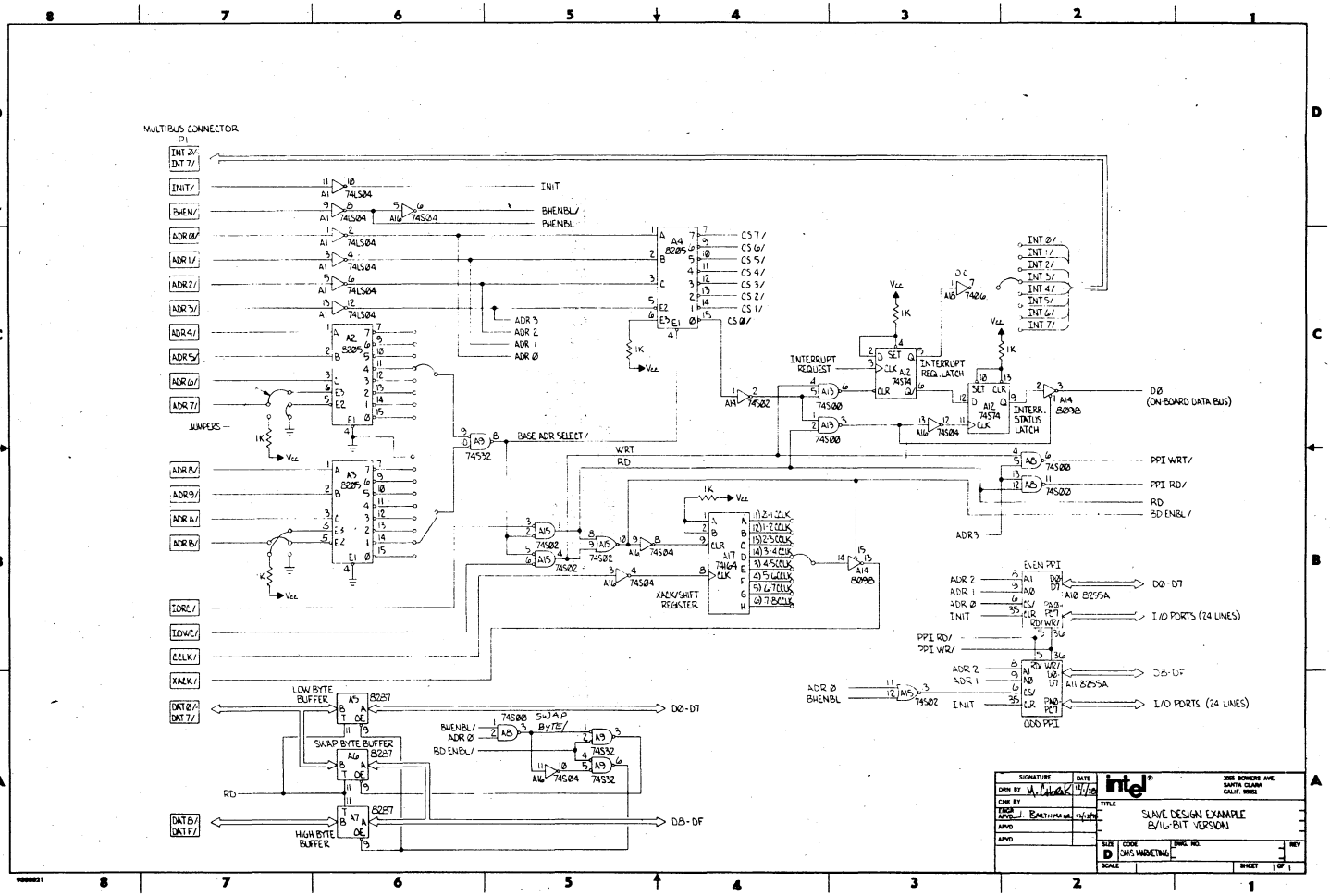
APPENDIX E
MECHANICAL SPECIFICATIONS



NOTES:

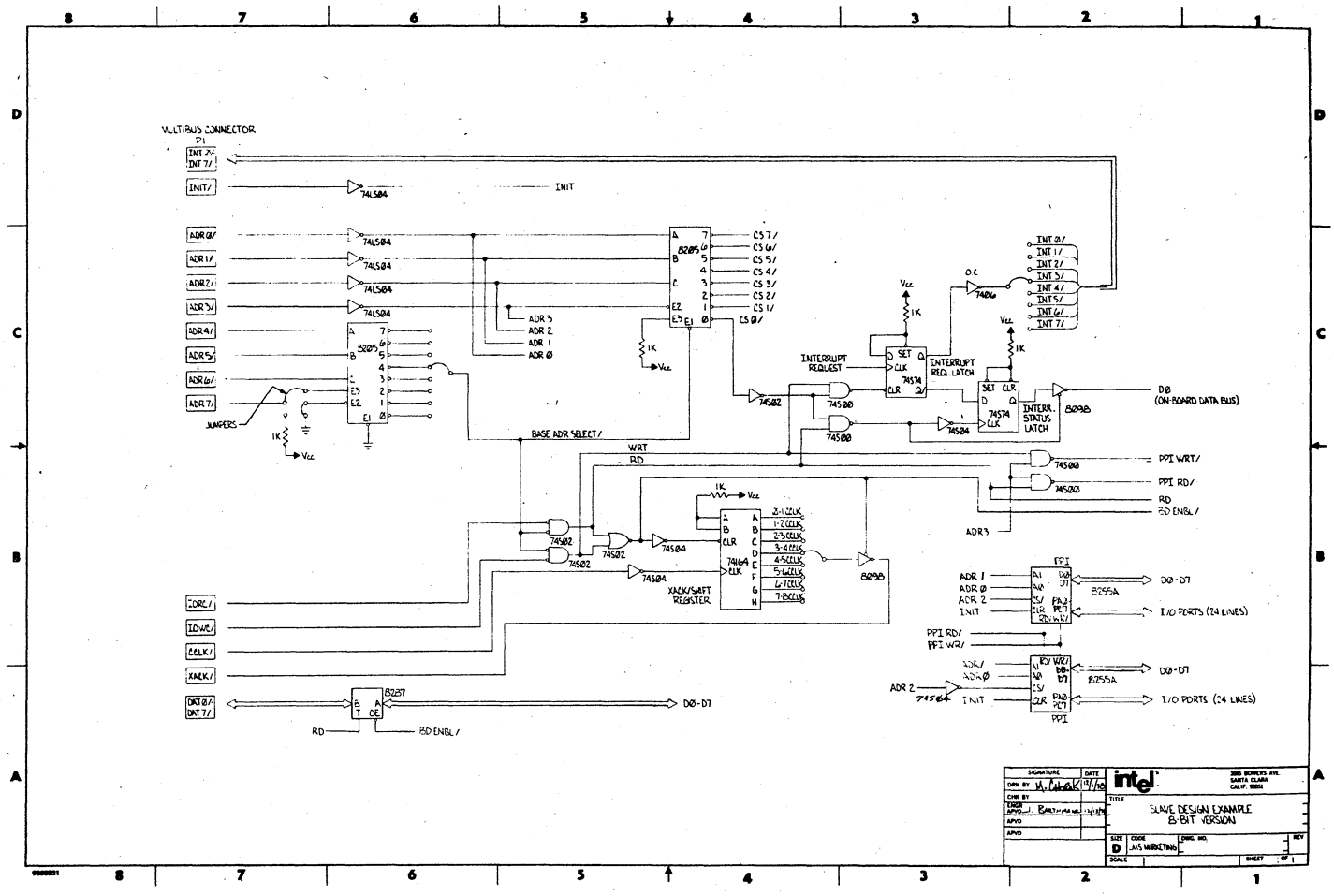
1. BOARD THICKNESS: 0.062
2. MULTIBUS® CONNECTOR: 86-PIN, 0.156 SPACING
3. AUXILIARY CONNECTOR: 60-PIN, 0.100 SPACING
4. EJECTOR TYPE: SCANBE #S203
5. BUS DRIVERS AND RECEIVERS SHOULD BE LOCATED AS CLOSE AS POSSIBLE TO THEIR RESPECTIVE MULTIBUS PIN CONNECTIONS
6. BOARD SPACING: 0.6
7. COMPONENT HEIGHT: 0.4
8. CLEARANCE ON CONDUCTOR NEAR EDGES: 0.050

APPENDIX F
MULTIBUS® SLAVE DESIGN EXAMPLE SCHEMATIC
8/16-BIT VERSION



MULTIBUS® SLAVE DESIGN EXAMPLE SCHEMATIC 8/16-BIT VERSION

APPENDIX G
MULTIBUS® SLAVE DESIGN EXAMPLE SCHEMATIC
8-BIT VERSION



MULTIBUS® SLAVE DESIGN EXAMPLE SCHEMATIC 8-BIT VERSION

APPENDIX B
BUS TIMING SPECIFICATIONS SUMMARY

Parameter	Description	Minimum	Maximum	Units
t_{BCY}	Bus Clock Period	100	D.C.	ns
t_{BW}	Bus Clock Width	0.35 t_{BCY}	0.65 t_{BCY} (Not Restricted)	
t_{SKEW}	BCLK/skew		3	ns
t_{PD}	Standard Bus Propagation Delay		3	
t_{AS}	Address Set-Up Time (at Slave Board)	50		ns
t_{DS}	Write Data Set Up Time	50		ns
t_{AH}	Address Hold Time	50		ns
t_{DHW}	Write Data Hold Time	50		ns
t_{DXL}	Read Data Set Up Time To XACK	0		ns
t_{DHR}	Read Data Hold Time	0	65	ns
t_{XAH}	Acknowledge Hold Time	0	65	ns
t_{XACK}	Acknowledge Time	0	8	μs
t_{CMD}	Command Pulse Width	100	9.5	ns
t_{ID}	Inhibit Delay	0	100 (Recommend < 100 ns)	ns
t_{XACKA}	Acknowledge Time of of an Inhibited Slave	t_{IAD} + 50 ns	1500	
t_{XACKB}	Acknowledge Time of an Inhibiting Slave	1.5	8	μs
t_{IAD}	Acknowledge Disable from Inhibit (An internal parameter on an inhibited slave; used to determine t_{XACKA} Min.)	0	100 (arbitrary)	ns
t_{AIZ}	Address to Inhibits High Delay		100	ns
t_{INTA}	INTA/ Width	250		ns
t_{CSEP}	Command Separation	100		ns

APPENDIX B (Continued)

BUS TIMING SPECIFICATIONS SUMMARY

Parameter	Description	Minimum	Maximum	Units
t_{BREQL}	↓BCLK/ to BREQ/ Low Delay	0	35	ns
t_{BREQH}	↓BCLK/ to BREQ/ High Delay	0	35	ns
t_{BPRNS}	BPRN/ to ↓BCLK/ Setup Time	22		ns
t_{BUSY}	BUSY/ delay from ↓BCLK/	0	70	ns
t_{BUSYS}	BUSY/ to ↓BCLK/ Setup Time	25		ns
t_{BPRO}	↓BCLK/ to BPRO/ (CLK to Priority Out)	0	40	ns
t_{BPRNO}	BPRN/ to BPRO/ (Priority In to Out)	0	30	ns
t_{CBRO}	↓BCLK/ to CBRQ/ (CLKto Common Bus Request)	0	60	ns
t_{CBRQS}	CBRQ/ to ↓BCLK/ Setup Time	35		ns
t_{XCD}	XACK↓ to Command↑ Delay	0	1500	ns
t_{BSYO}	CBRQ/↓ and BUSY/↓ to BUSY/↑	—	12	μs
t_{CCY}	C-clock Period	100	110	ns
t_{CW}	C-clock Width	0.35 t_{CCY}	0.65 t_{CCY}	ns
t_{INIT}	INIT/Width	5		ms
t_{INITS}	INIT/ to MPRO/ Setup Time	100		ns
t_{PBD}	Power Backup Logic Delay	0	200	ns
t_{PFINW}	PFIN/ Width	2.5		ms
t_{MPRO}	MPRO/ Delay	2.0	2.5	ms
t_{ACLOW}	ACLO/ Width	3.0		ms
t_{PFSRW}	PFSR/ Width	100		ns
t_{TOUT}	Timeout Delay	5	∞ (D.C.)	ms
t_{DCH}	D.C. Power Supply Hold from ACLO/	3.0		ms
t_{DCS}	D.C. Power Supply Setup to ACLO/	5		ms

APPENDIX C
BUS DRIVERS, RECEIVERS, AND TERMINATIONS

Driver 1,3						Receiver 2,3				Termination			
Bus Signals	Location	Type	I_{OL} Min_{ma}	I_{OH} $Min_{\mu a}$	C_O Max_{pf}	Location	i_{IL} Max_{ma}	i_{IH} $Max_{\mu a}$	C_I Max_{pf}	Location	Type	R	Units
DAT0/→DATF/ (16 lines)	Masters and Slaves	TRI	16	-2000	300	Masters and Slaves	-0.8	125	18	1 place	Pullup	2.2	KΩ
ADR0/-ADRB/, BHEN/ (21 lines)	Masters	TRI	16	-2000	300	Slaves	-0.8	125	18	1 place	Pullup	2.2	KΩ
MRDC/,MWTC/	Masters	TRI	32	-2000	300	Slaves (Memory; memory-mapped I/O)	-2	125	18	1 place	Pullup	1	KΩ
IORC/,IOWC/	Masters	TRI	32	-2000	300	Slaves (I/O)	-2	125	18	1 place	Pullup	1	KΩ
XACK/	Slaves	TRI	32	-2000	300	Masters	-2	125	18	1 place	Pullup	510	Ω
INH1/,INH2/	Inhibiting Slaves	OC	16	—	300	Inhibited Slaves (RAM, PROM, ROM, Memory-Mapped I/O)	-2	50	18	1 place	Pullup	1	KΩ
BCLK/	1 place (Master us)	TTL	48	-3000	300	Master	-2	125	18	Mother-board	To +5V To GND	220 330	Ω Ω
BREQ/	Each Master	TTL	5	-200	60	Central Priority Module	2	50	18	Central Priority Module (not req)	Pullup	1	KΩ
BPRO/	Each Master	TTL	5	-200	60	Next Master in Serial Priority Chain at its BPRN/	-1.6	50	18	(not req)			
BPRN/	Parallel: Central Priority Module Serial:Prev Masters BPRO/	TTL	5	-200	300	Master	-4	100		(not req)			
BUSY/, CBRQ	All Masters	O.C.	20	—	300	All Masters	-2	50	18	1 place	Pullup	1	KΩ
INIT/	Master,	O.C.	32	—	300	All	-2	50	18	1 place	Pullup	2.2	KΩ
CCLK/	1 place	TTL	48	-3000	300	Any	-2	125	18	Mother-board	To +5V To GND	220 330	Ω Ω
INTA/	Masters	TRI	32	-2000	300	Slaves (Interrupting I/O)	-2	125	18	1 place	Pullup	1	KΩ
INT0/→INT7/ (8 lines)	Slaves	O.C.	16	—	300	Masters	-1.6	40	18	1 place	Pullup	1	KΩ
PFSR/	User's Fron Panel?	TTL	16	-400	300	Slaves, Masters	-1.6	40	18	1 place	Pullup	1	KΩ
PFSN/	Power Back Up Unit	TTL	16	-400	300	Masters	-1.6	40	18	1 place	Pullup	1	KΩ
ACLO	Power Supply	O.C.	16	-400	300	Slaves, Masters	-1.6	40	18	1 place	Pullup	1	KΩ
PFIN/	Power Back-Up Unit	O.C.	16	-400	300	Masters	-1.6	40	18	1 place	Pullup	1	KΩ
MPRO/	Power Back-Up Unit	TTL	16	-400	300	Slaves Masters	-1.6	40	18	1 place	Pullup	1	KΩ

APPENDIX C (Continued)

BUS DRIVERS, RECEIVERS, AND TERMINATIONS

Driver 1,3						Receiver 2,3				Termination			
Bus Signals	Location	Type	I_{OL} Min_{ma}	I_{OH} $Min_{\mu a}$	C_O Max_{pf}	Location	I_{IL} Max_{ma}	I_{IH} $Max_{\mu a}$	C_I Max_{pf}	Location	Type	R	Units
Aux Reset/	User's Front Panel?	Switch to GND (Note 5)	—	—	—	Masters	-2	50	18	None			

Notes:

1. Driver Requirements

   I_{OH} = High Output Current Drive
   I_{OL} = Low Output Current Drive
   C_O = Capacitance Drive Capability
   TRI = 3-State Drive
   O.C. = Open Collector Driver
   TTL = Totem-pole Driver

2. Receiver Requirements

   I_{IH} = High Input Current Load
   I_{IL} = Low Input Current Load
   C_I = Capacitive Load

3. TTL low state must be $\geq$ -0.5v but $\leq$ 0.8v at the receivers
   TTL high state must be $\geq$ 2.0v but $\leq$ 5.5v at the receivers
4. For the iSBC 80/10 and the iSBC 80/10A use only a 1K pull-up resistor to +5v for BCLK/ and CCLK/ termination.
5. Recommend a 47Ω resistor in series with switch.

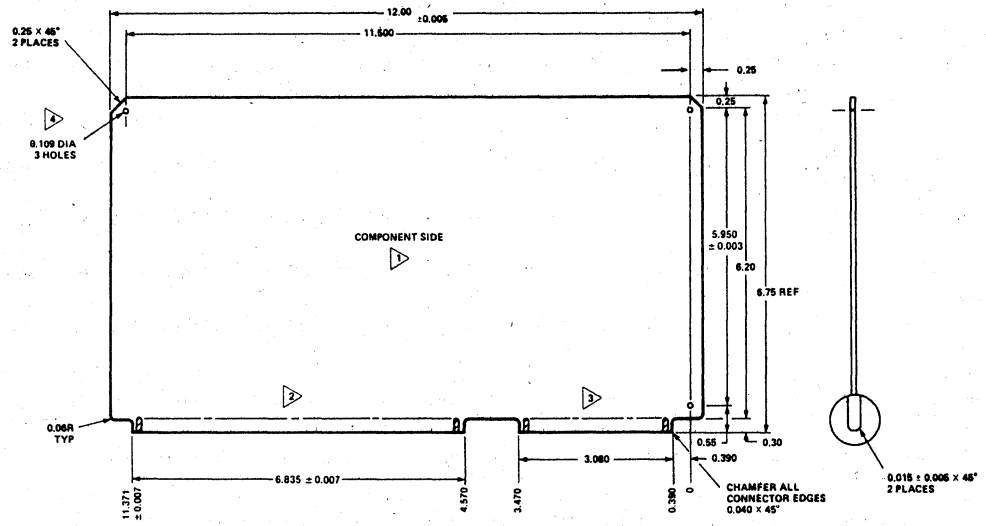
APPENDIX D
BUS POWER SPECIFICATIONS

	Standard (P1)				Optional (P2)					
					Analog Power		Battery Power Backup			
	Ground	**+5**	**+12**	**−12**	**+15**	**−15**	**+5**	**+12**	**−12**	**−5**
Mnemonic	GND	+5V	+12V	−12V	+15V	−15V	+5B	+12B	−12B	−5B
Bus Pins	P1+1,2, 11,12, 75,76 85,86	P1+3,4, 5,6,81, 82,83, 84	P1+7,8	P1+79, 80	P2+23, 24	P2+25, 26	P2+3,4, 5,6	P2+11, 12	P2+15, 16	P2−7,8
Nominal Output	Ref.	+5.0V	+12.0V	−12.0V	+15.0V	−15.0V	+5.0V	+12.0V	−12.0V	−5.0V
Tolerance from Nominal[1]	Ref.	±5%	±5%	±5%	±3%	±3%	±5%	±5%	±5%	±5%
Ripple (Pk-Pk)[2]	Ref.	50 mV	50 mV	50 mV	10 mV	10 mV	50 mV	50 mV	50 mV	50 mV
Transient Response Time[3]		500 μs	500μs	500 μs	100 μs	100 μs	500 μs	500 μs	500 μs	500 μs
Transient Deviation[4]		±10%	±10%	±10%	±10%	±10%	±10%	±10%	±10%	±10%

NOTES:

1. Tolerance is worst case, including initial voltage setting line and load effects of power source, temperature drift, and any additional steady state influences.
2. As measured over any bandwidth not to exceed 0 to 500 kHz.
3. As measured from the start of a load change to the time an output recovers within ±0.1% of final voltage.
4. Measured as the peak deviation from the initial voltage.

APPENDIX E
MECHANICAL SPECIFICATIONS



NOTES:

1. BOARD THICKNESS: 0.062
2. MULTIBUS® CONNECTOR: 86-PIN, 0.156 SPACING
3. AUXILIARY CONNECTOR: 60-PIN, 0.100 SPACING
4. EJECTOR TYPE: SCANBE #S203
5. BUS DRIVERS AND RECEIVERS SHOULD BE LOCATED AS CLOSE AS POSSIBLE TO THEIR RESPECTIVE MULTIBUS PIN CONNECTIONS
6. BOARD SPACING: 0.6
7. COMPONENT HEIGHT: 0.4
8. CLEARANCE ON CONDUCTOR NEAR EDGES: 0.050

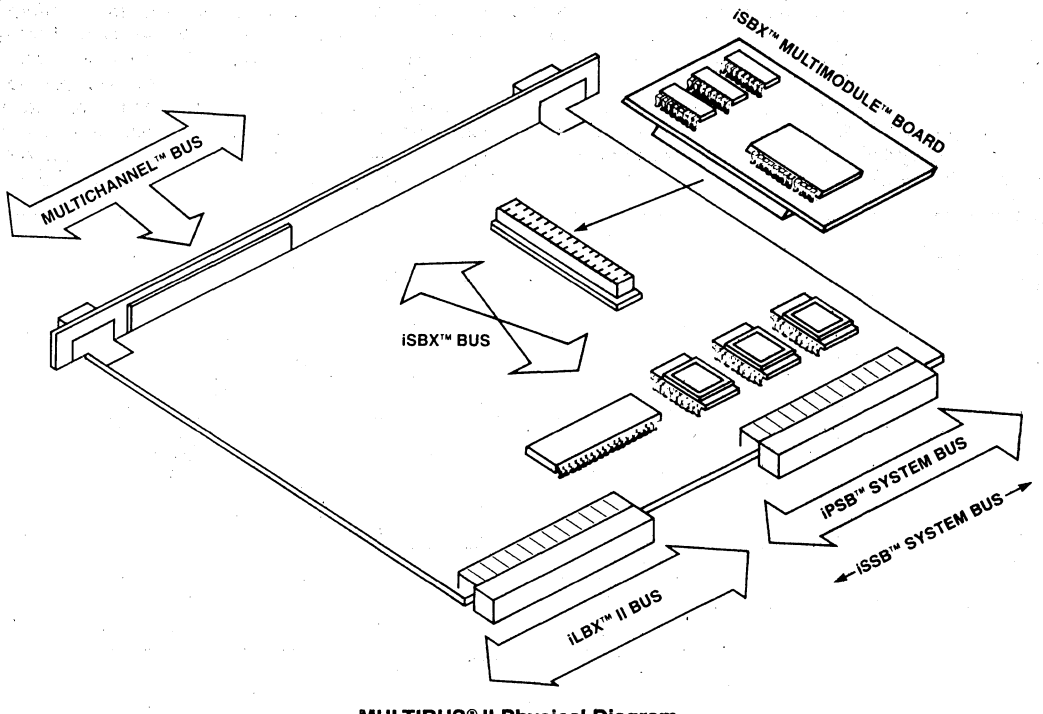
MULTIBUS® II Architecture and Products

4

MULTIBUS® II
iLBX™ II Local Bus Extension

- **High bus bandwidth —**
  — 48 megabytes/sec
- **64 megabyte (26-bit) addressing**
- **8-, 16-, 24-, and 32-bit data transfers over a 32-bit path**
- **Reliable synchronous clocking up to 12 megahertz**
- **Burst transfers up to 64 kilobytes per transfer**
- **Primary and secondary bus master exchange capabilities**
- **Supports up to 6 iLBX™ II compatible devices per bus**
- **Pipelined protocol for highest performance**
- **Optional parity protection for address and data**

The iLBX™ II Local Bus Extension is one of the family of standard bus structures resident within Intel's MULTIBUS® II Bus Architecture. The iLBX II bus is a dedicated execution bus capable of significantly increasing system performance by removing most processor execution activity from the main iPSB™ Parallel System Bus. It extends the processor board's on-board local bus to off-board resources. Acting in conjunction with the processor board, the iLBX II resources form a multiple board "virtual single board computer". The iLBX II bus preserves advantages in performance and architecture of on-board local memory, while allowing memory configurations larger than those possible on a single board.



MULTIBUS® II Physical Diagram

October, 1984
Order Number: 230893-002

FUNCTIONAL DESCRIPTION

Architectural Overview

The iLBX II bus is an architectural solution for supporting large amounts of off-board memory with the same performance advantage enjoyed by on-board memory (see Figure 1). It allows the CPU board selection to be decoupled from the on-board memory requirement and still maximizes the processor's performance potential. It eliminates the processor's need to access its off-board memory resources solely over the iPSB system bus. In most systems, the processor is the only master on the iLBX II bus, so no time is required to arbitrate for the bus. This means the processor sees significantly lower memory latency than is possible if it were accessing memory over the multiple master system bus. Lower memory latency translates to higher individual processor performance.

The inclusion of the iLBX II bus in the architecture means not just higher single processor performance but higher system performance as well. The movement of execution traffic from the system bus to the iLBX II execution bus makes that much additional system bus bandwidth available to other system resources such as processors not using an execution bus or I/O devices.

For those applications which require a high bandwidth local path to I/O, such as an intelligent disk controller local to a particular processor, the iLBX II bus supports one additional bus master. This architectural enhancement allows a processor to "own" an intelligent I/O controller. All data transfers between these two modules (the processor and the controller) can occur over the low latency iLBX II bus path without disturbing activity on the system bus.

Structural Features

Overview

The iLBX II bus uses a non-multiplexed, processor independent structure supporting 8-, 16-, and 32-bit processors. It supports 8-, 16-, 24-, and 32-bit data transfers over a 26-bit (64 megabyte) addressing range with a maximum bandwidth of 48 megabytes/sec.

All events performed on the bus are synchronous to a reference bus clock. This is not a fixed frequency clock as in the iPSB bus; the iLBX II bus clock runs at the basic processor bus frequency. In other words, a processor whose bus interface runs at 8 megahertz would drive the iLBX II bus at that frequency. This characteristic helps match the iLBX II bus timing to that of the processor transfer rate for best performance. The maximum iLBX II bus clock frequency is 12 megahertz. (Be careful not to confuse a processor's clock input frequency with its basic bus frequency. Many processors internally divide down their clock input by 2, 3, or 4 to obtain the basic bus frequency. It is this basic bus frequency which defines their transfer rate and which drives the iLBX II bus clock.)

Non-Multiplexed Structure

The iLBX II bus structure is non-multiplexed in order to simplify the interface and obtain maximum performance. The separate address, data, and control paths allow overlapped operation. This overlapping, called pipelining, means that data from a previous operation can be overlapped with the address and command information of the current operation. This characteristic substantially improves bus utilization for those processor-memory subsystems which support the feature.

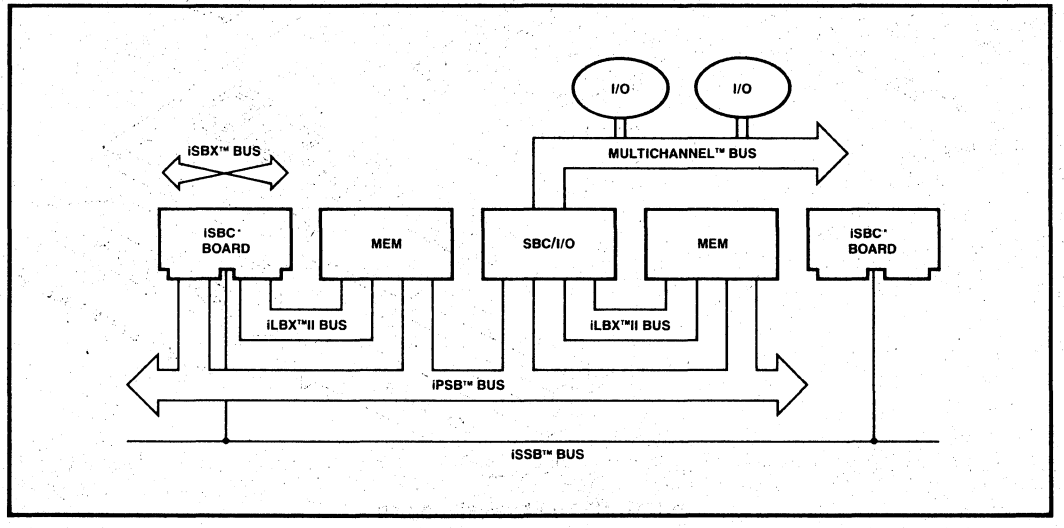


Figure 1. MULTIBUS® II Bus Architecture

Interconnect Address Space

The iLBX II bus supports the slot-addressing concept of the interconnect address space found in the iPSB bus. Including this facility in the iLBX II bus allows the system to identify and configure iLBX II bus boards even though they may not contain a iPSB bus port. (Please refer to the iPSB bus data sheet for additional information on the Interconnect address space.)

Dual Bus Masters

In order to support a wide range of system configurations, the iLBX II bus defines support for two bus masters. One master is called the Primary master; the other is known as the Secondary master. The Primary master normally "owns"the bus and does not have to spend any time arbitrating for access rights. The Secondary master must ask the Primary master for access rights. The Primary releases the bus at the first opportune time. This hierarchical structure ensures that the Primary master enjoys good memory latency while at the same time gives the Secondary the opportunity to access memory when it needs to.

The iLBX II bus also includes a dedicated interrupt line to facilitate signalling between the two masters for commands and status, and between the memory boards and the Primary master for things such as non-recoverable memory errors.

Bus Cycle Overview

Like the iPSB bus, the iLBX II bus protocol consists of three types of bus cycles: arbitration, transfer, and exception.

Arbitration Cycle

The arbitration cycle ensures that one and only one requesting agent is allowed access to the bus at any given time. When a requesting agent determines the need for a bus operation, it enters the arbitration cycle. For either requesting agent, this cycle lasts until it acquires the right to use the bus. In configurations with only a primary requesting agent, no time is spent for this cycle; the agent always has rights to the bus. In configurations where there are both a primary and secondary agent, the primary agent has to arbitrate for the bus only when the bus is busy under the secondary agent's control. Figure 2 illustrates the arbitration cycle.

Transfer Cycle

The transfer cycle is the event where the request (address and command) and reply (data) information is exchanged between the bus agents. Like the iPSB bus, it consists of a request and a reply phase. During block transfers, the termination of the transfer cycle is controlled by the requesting agent. In non-block transfer cycles, the cycle's termination is implicitly recognized by both agents. Figure 3 shows a transfer cycle example.

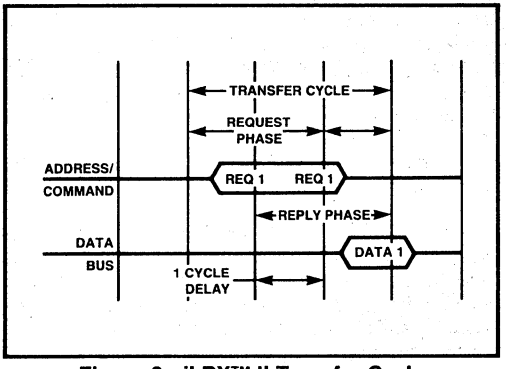


Figure 3. iLBX™ II Transfer Cycle

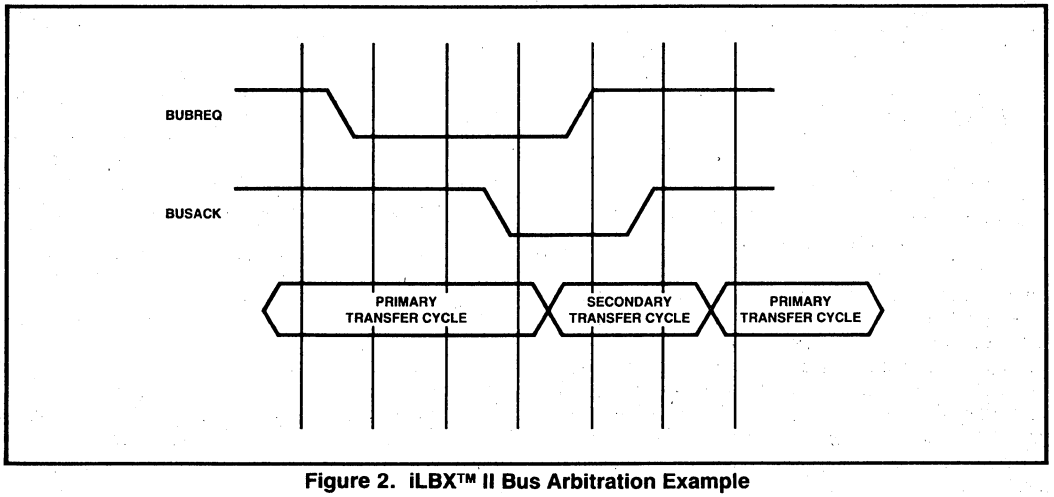


Figure 2. iLBX™ II Bus Arbitration Example

Exception Cycle

Exception cycles allow the bus agents to signal any detected error or exceptional condition which might arise during a transfer cycle. Typical exceptions are uncorrectable ECC errors, parity errors, or physical boundary overflows.

Signal Groups

Overview

There are five categories of signals used in the iLBX II bus: address/command, data transfer, access control/status, bus control/status, and miscellaneous. An asterisk following the signal name or group indicates that the signal or group use their low electrical state as the active state.

Address/Command

The requesting agent uses this group of signals to transfer address and command information to the potential replying agents during the request phase of a transfer cycle. This signal group consists the non-multiplexed address lines, **XA25 through XA00** (Extension bus address), the command specification lines, **XC3 through XC0** (Extension bus command), and an associated parity line, **XAPAR** (Extension bus address/command parity).

The XA25 through XA00 lines define the starting physical byte address. The command specification lines select the address space (memory or interconnect), data width (1, 2, 3, or 4 bytes), and whether the operation is a read or write cycle. The command encodings for XC3 through XC0 are shown in Figure 4.

XC3*	XC2*	XC1*	XC0*
Address Space	**Access Type**	**Width Specification**	
Memory	Read	1 byte 2 bytes	
Interconnect	Write	3 bytes 4 bytes	

Figure 4. iLBX II Command Encoding

Parity for the address/command group is not required. The bus does allow for a single parity bit covering the address and command lines as a compliance level. The iLBX II bus environment is much different than that of the iPSB system bus. It extends only a short distance (6 card slots maximum) and employs lower switching currents. This more restrictive environment reduces the need for data integrity protection in all but the larger systems.

Data Transfer Group

This signal category consists of the 32 bi-directional data lines and their optional parity line. **XD31 through XD0** (Extension bus data) transfer the read or write data between the requesting and replying agents. Each byte in the iLBX II bus memory is mapped to one of the four byte locations of the XD lines. This technique is commonly referred to as "byte lanes" and is illustrated in Figure 5.

Like with the address/command group, the **XDPAR** (Extension bus data parity) line is optional.

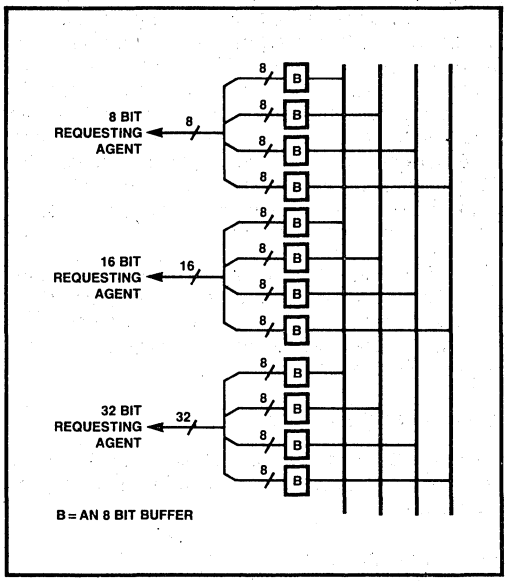


Figure 5. iLBX™ II Data Bus Alignment Interface Requirements

Access Control/Status Group

This signal category consists of 5 lines which determine the start of an access request, its execution, and finally, its termination.

The **XACCREQ*** (Extension bus access request) signal indicates that the address/command information is valid during the current and next bus clock cycles. It signals the presence of the request phase of the transfer cycle. Replying agents which require more time to decode the command information can extend XACCREQ* using the XWAIT* handshake line.

The **XWAIT*** (Extension bus wait) signal has a two-fold meaning in the access protocol: it can extend the duration of the request phase and it serves as a "not ready" replier indication during the reply phase. If asserted in the first clock cycle of the request phase, it extends the phase, otherwise, it will signal "not ready" during the reply phase.

In many system configurations the iLBX II bus memory boards are dual-ported to both the iLBX II and iPSB buses. This requires a mutual exclusion facility when implementing semaphores and other data structures in this shared memory. The **XLOCK*** (Extension bus lock) signal allows the iLBX II bus requesting agents to lock out the other port while performing indivisible accesses to shared structures.

To perform block transfers on the iLBX II bus, the requesting agent asserts the **XBTCTL*** (Extension bus block transfer control) signal. This line informs the replying agents that two or more data transfer periods will accompany a single request phase. XBTCTL* is de-asserted by the requesting agent to signal the end of the block transfer.

Bus Control/Status Group

The signals in this group control the passing of bus ownership between the primary and secondary requesting agents. When the bus is in use, they also indicate which agent is in control.

The **XBUSREQ*** (Extension bus request) signal is driven by the secondary requesting agent to acquire the bus from the primary agent. Only the primary requesting agent receives this signal. When the primary detects that the secondary is requesting the bus, it replies with the **XBUSACK*** (Extension bus acknowledge) signal to inform the secondary that the bus is now his. This bus exchange occurs at the discretion of the primary.

The secondary owns the bus after asserting XBUSREQ* and receiving XBUSACK* active. The primary can request that the bus be returned at any time by removing XBUSACK*. The secondary must return the bus at the earliest time; typically when it completes its current transfer cycle.

Miscellaneous Control Group

The **XRESET*** (Extension bus reset) is driven by the primary requesting agent to locally initialize its iLBX II bus environment. It is typically asserted after the agent receives a reset indication on the iPSB system bus.

The **XINT*** (Extension bus interrupt) allows the secondary requesting agent and any of the replying agents to signal the primary requesting agent for inter-module communication. Since the secondary agent is usually performing tasks on behalf of the primary agent, this interrupt line removes the need for the primary to continuously poll the secondary for completion of its tasks.

The **XID2* through XID0*** (Extension bus identity) lines are hardwired lines on the backplane to allow any iLBX II bus board to determine its position on the bus. They encode the interconnect space least significant three bits of the slot ID field. (See the iPSB bus data sheet for an explanation of the interconnect address space.)

The final line is the **XBCLK*** (Extension bus clock) line. It provides the reference timing signal for the synchronous bus operations. It is driven by the primary requesting agent at its processor bus frequency.

The iLBX II bus also defines additional +5 volt and ground pins.

Bus Protocol

In the MULTIBUS II specification, both timing diagrams and state-flow diagrams describe the iLBX II bus protocol. The state-flow diagrams present the lowest level and most rigorous definition while the timing diagrams help conceptual understanding. For the purposes of this data sheet, only the timing diagram description is used. The following sections use Figure 6 as an example of the protocol.

Arbitration Cycle

With only two potential requesting agents contending for access rights to the bus, the arbitration cycle is very simple. The figure illustrates the secondary requesting agent requesting the bus from the primary and then running a simple transfer cycle. The secondary requesting agent makes its request by asserting XBUSREQ*. The primary gives up the bus by returning XBUSACK* active. In this example, the secondary uses the bus for only a single transfer cycle so it de-asserts XBUSREQ* when complete. The primary agent responds by withdrawing XBUSACK* to indicate it now owns the bus.

Transfer Cycle

Like in the iPSB bus, the transfer cycle proceeds as a request phase and a reply phase. The requesting agent (either the primary or the secondary depending upon who currently owns the bus) informs the potential replying agents of the request phase by driving valid information on the address/command signal group and asserting XACCREQ*. The request phase normally lasts two clock cycles although the replying agents have the opportunity to extend the phase as long as necessary by asserting XWAIT* during the first clock period of the phase. The phase is extended as long as XWAIT* is active. In the example, the request phase is extended one additional clock.

The reply phase begins when XWAIT* is de-asserted. At this point, the meaning of XWAIT* changes to become a "not ready" indication from the selected replying agent. In the example, the replying agent requires one additional clock period to supply the data so XWAIT* is asserted for one clock. The reply phase terminates on the same clock that data is valid.

Exception Cycle

If transfer integrity checking is implemented on the iLBX II bus, errors are signalled on the clock following the last valid information period. In the example, errors

detected on the address/command lines during the request phase are signalled on the clock following the removal of valid request information. The same applies to errors detected on the data lines during the reply phase.

Mechanical

The iLBX II bus is defined on the P2 connector of two-connector MULTIBUS II boards. Since the iLBX II bus environment is local to a particular processor board, the iLBX II bus backplane does not extend the entire length of the iPSB bus backplane. This allows for multiple iLBX II bus environments in a given system.

The pin assignment for the iLBX II bus on P2 is shown in iLBX II specification section in the MULTIBUS II Bus Architecture Specification Handbook.

Please refer to Intel's MULTIBUS II Bus Architecture Specification Handbook for more detailed information.

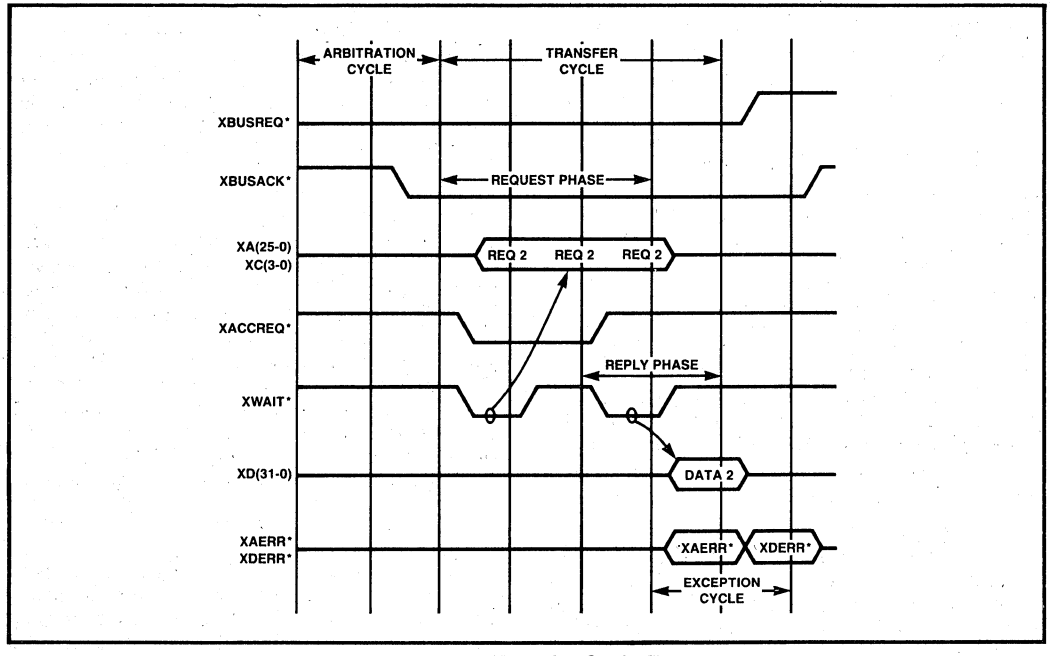
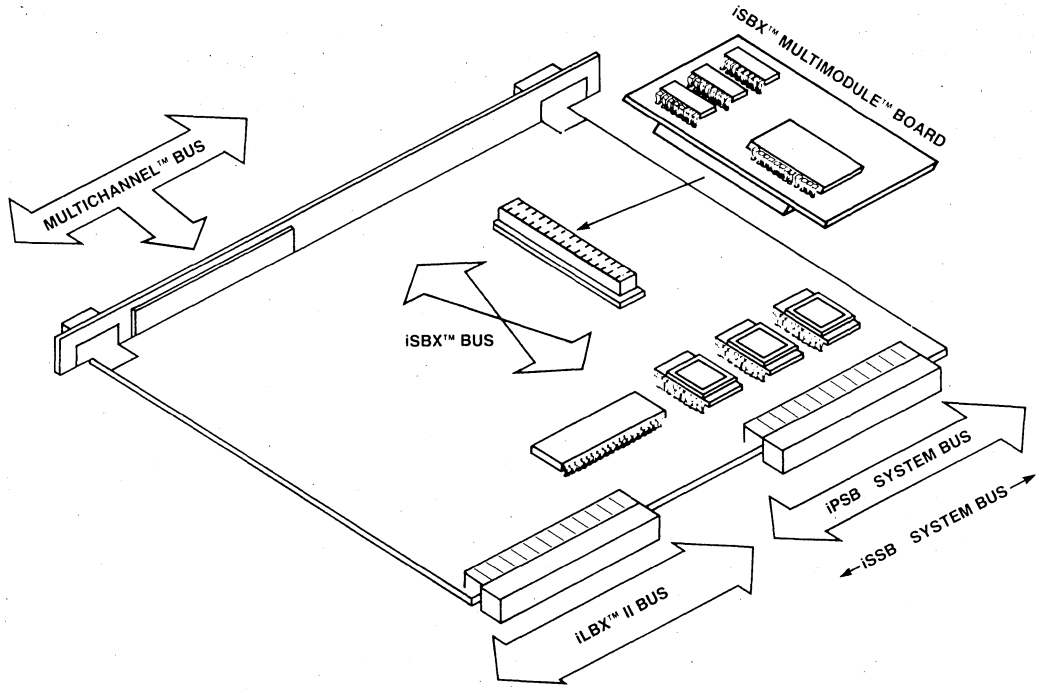


Figure 6. iLBX™ Transfer Cycle Example

MULTIBUS® II
iPSB Parallel System Bus

- **Very high bandwidth —**
  - **— 40 megabytes/sec using burst transfers**
  - **— 20 megabytes/sec with single cycles**
- **4 gigabyte (32-bit) addressing**
- **8-, 16-, 24-, and 32-bit data transfers over a 32-bit path**
- **Pin-efficient multiplexed structure**
- **Reliable synchronous clocking at 10 megahertz with full handshaking for data**
- **Distributed arbitration with up to 20 bus masters**
- **Full parity protection for data transfer integrity**
- **Message passing facility for inter-module communication**
- **Interconnect facility for software identification and configuration of boards**
- **Industry standard Eurocard form factors — 233mm x 220mm and 100mm x 220mm**

The MULTIBUS® II iPSB Parallel System Bus is the foundation of the MULTIBUS II Bus Architecture. It is a general-purpose, processor independent structure which fully supports 8-, 16-, and 32-bit microprocessors. This very high bandwidth structure is defined on a single 96-pin IEC 603-2 (DIN) connector. All data movement functions required in a microcomputer system are defined including such advanced functions as an integrated message passing protocol and an interconnect facility which allows software to address a board by its slot position for software-based board identification and configuration.



MULTIBUS® II Physical Diagram

October, 1984
Order Number: 230893-002

FUNCTIONAL DESCRIPTION

Architectural Overview

The MULTIBUS II iPSB Parallel System Bus is the foundation of the MULTIBUS II bus architecture (see Figure 1). As a system bus, it is a very high bandwidth (40 megabytes/sec) bus optimized for inter-module communication, however, it also defines the complete set of basic bus functions required in a microcomputer system: memory accesses for execution or data, accesses to I/O for control of I/O functions, plus inter-module signalling. These basic functions are supplemented with additional functions supporting geographical (by-slot) addressing and an integral message passing protocol.

Geographical addressing allows addressing of individual boards via their physical position in the backplane. Software can determine what boards are being used and configure itself appropriately. Software also can configure the hardware characteristics of the board (eg. the starting address of a memory board). This can substantially reduce or even eliminate hardware jumper options and DIP switches for board configuration. Geographical addressing is a function of the interconnect address space.

MULTIBUS II's integral message passing protocol defines a standard and uniform way for modules to communicate over either the iPSB or iSSB™ buses. Integrating the protocol at the bus structure level lets the designer provide hardware support to increase system inter-module communication performance and opens the door for VLSI solutions. Standardizing the interface ensures a uniform software interface so that users can take advantage of new advances in technology without having to rewrite software.

Structural Features

Overview

The iPSB bus structure is a processor-independent general-purpose bus designed to support 8-, 16-, and 32-bit processors. It is designed to operate at a maximum bandwidth of 40 megabytes/sec while using off-the-shelf components.

Special attention has been given to how the bus structure, both electrically and mechanically, impacts system reliability. Synchronous sampling of all bus signal lines assures good immunity from crosstalk and noise. Full byte parity generation and checking protects all transfers on the bus to ensure that any bus error is detected. And a signal quality on the bus is excellent due to the large number of interlaced ground lines. Mechanically, the iPSB bus is defined on a two-piece 96-pin IEC 603-2 to ensure good connector reliability.

Multiplexing

The iPSB bus is highly multiplexed. The 32-bit address and data paths are multiplexed and the nine system control lines have different uses depending upon the phase of the transfer cycle. The six arbitration lines also serve dual purposes between system initialization and normal operation.

This multiplexed structure has several benefits. First of all, the entire 32-bit iPSB bus is defined on a single connector. This allows a full 32-bit iPSB bus interface on even the smaller, single connector, form factor board. This opens the possibility of low cost 32-bit systems. Multiplexing also reduces by half the number of high current drivers required for the interface which significantly reduces a board's current requirements. The routing of signal lines between the bus interface and connector is simplified.

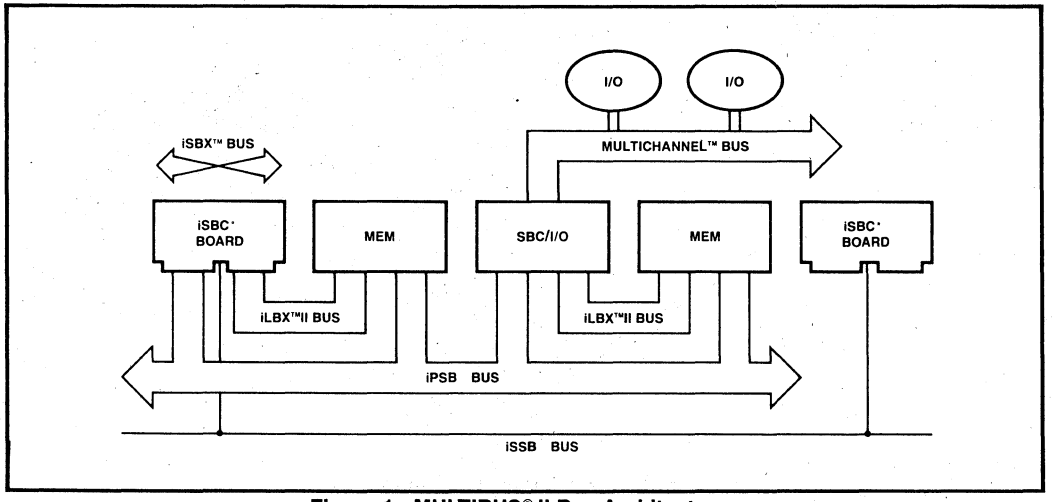


Figure 1. MULTIBUS® II Bus Architecture

Bus Errors and Exceptions

The iPSB bus defines a complete set of bus error reporting mechanisms. Serious errors such as a parity error or the failure of a module to complete the data handshake, are flagged on unique bus signal lines and are seen by all modules on the bus. These errors induce a recovery time in which the bus is allowed to stabilize before further transfer cycles may begin.

The iPSB bus also provides mechanisms for signaling less serious operational errors. Operational errors, such as attempting to perform a 32-bit access to a 8-bit device or writing to read-only memory, are signaled as agent exceptions. These exceptions may induce retry operations by an intelligent bus interface or may be passed to the offending processor as errors.

Interconnect Address Space

The ability to address a board by its physical position in the backplane is also supported in the iPSB bus. This facility allows board manufacturers to code such items as their vendor number, board type, board revision number, and serial number on the board. This information is available to the system software. This facility is defined in the iPSB bus interconnect address space.

Aside from this read-only information, the interconnect space allows write operations to support board configuration and diagnostics under software control. This facility can help reduce or eliminate hardware-based jumper options and DIP switches.

Interrupts

The iPSB bus supports up to 255 distinct interrupt sources and 255 interrupt destinations. Rather than the use of the traditional method of dedicated interrupt signal lines on the bus, the iPSB bus defines a special bus cycle to convey interrupt information. This special bus cycle (actually part of the message passing protocol discussed below) redefines the meaning of the address; instead of a byte location in memory for example, 16 of the 32 lines encode 8 bits for the source module generating the interrupt and 8 bits for the destination module to service the interrupt.

This technique overcomes the significant problem of interrupt configuration found in traditional buses. Dedicated lines usually imply that only one particular destination can service one particular interrupt source. If an interrupt source wishes to target some interrupts to one destination and some to a different destination, separate bus interrupt lines are required for each destination. This can quickly consume all dedicated interrupt lines in even a moderate size system.

Using interrupt bus cycles with embedded source and destination module addressing removes the need for dedicated interrupt lines at the same time it allows any interrupt source to signal any interrupt destination.

Message Passing

With the trend in microcomputer systems toward multiprocessing, it is important to provide the facilities and mechanisms to lend support for inter-module communication. The iPSB bus includes such mechanisms and defines the protocol for greatly enhanced performance in inter-module communication. This protocol is called MULTIBUS II Message Passing.

Most multiprocessor systems use either a "pass by reference" or a "pass by value" protocol for intermodule communication. In the "pass by reference" case, the two modules share a common memory resource and pass pointers or tokens to extend addressability of a desired data structure to the other module. In "pass by value", the modules exchange a copy of the desired data structure. Each of these protocols has a set of advantages and disadvantages associated with performance, data security, extensibility to additional modules, and ease of use.

MULTIBUS II Message Passing takes the best of both methods and lends hardware support. Message passing uses a hardware "pass by value" interface that gives the performance of a "pass by reference" system. It replaces the software module used by the "pass by value" method with a specialized message passing interface. The processor "passes by reference" the reference to the data structure to the message passing interface. This interface communicates with the destination module's message passing interface to transfer the data without processor intervention. This data transfer is performed in the message address space. This is illustrated in Figure 2. (In many ways, it is helpful to think of the two communication message passing interfaces as a distributed, smart, DMA controller.)

There are several significant benefits to this approach. First of all, the message passing interfaces can take advantage of the full capabilities of the bus (ie, 32-bit data and burst transfer) independent of the type or nature of the controlling processor. Even 8-bit processor or I/O boards can take full advantage of the bus. This means significantly higher inter-module communication performance over a completely software-base method. Another benefit is the elimination of any shared memory. Dual-ported memory structures are no longer needed nor are global memory boards. The other primary benefit is that MULTIBUS II message passing presents a uniform software interface for all modules. Modules can be replaced with new modules containing newer technology (e.g. moving from a single density to a double density disk controller) without any software changes required in the controlling module. This makes it easy for users to integrate new technology without the problem of completely rewriting the driver software.

Central Services Module

The iPSB bus specification defines the central system functions as the Central Services Module (CSM). The minimal set of functions are: clock generation, power-down and reset, time-out, and assignment of slot IDs. Collecting these functions in a single module improves overall board area utilization, since the functions are not duplicated on every board and then only used on one. The system designer is free to implement the CSM on a separate board or to include the functions as just one of several modules on another board.

Bus Cycle Overview

The iPSB bus defines three types of bus cycles: arbitration, transfer, and exception cycles. Each cycle is made up of one or more phases. Figure 3 illustrates the relationship among these cycles and phases.

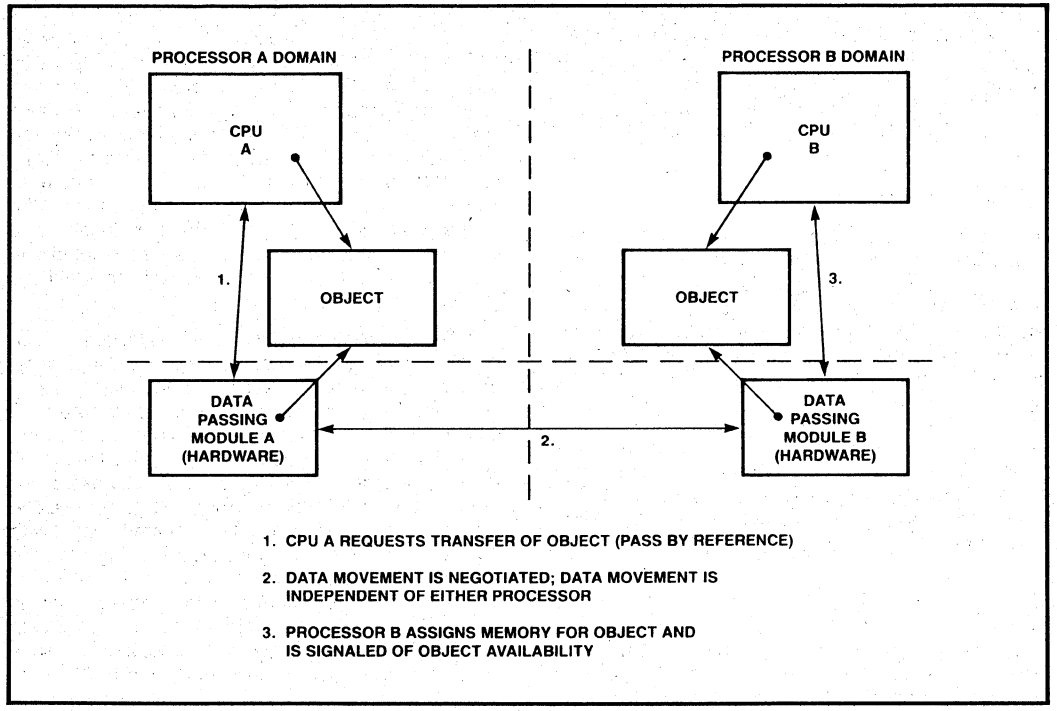


Figure 2. MULTIBUS® II Message Passing

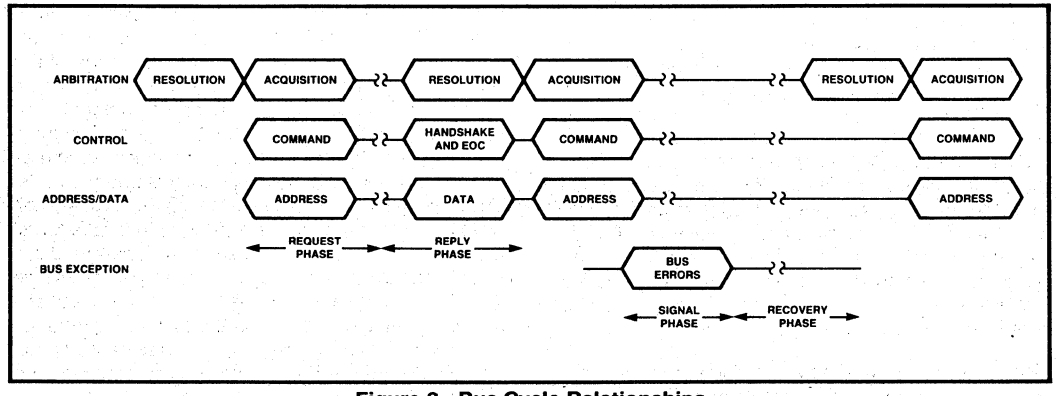


Figure 3. Bus Cycle Relationships

Arbitration Cycle

The arbitration cycle is made up of a resolution phase and an acquisition phase. The resolution phase is the time-period in which all agents collectively arbitrate for access rights to the bus. Depending on the arbitration algorithm, the agents decide among themselves which of them is going to control the bus after the current bus owner is done. This arbitration method is referred to as self-selecting since the agents decide ownership among themselves.

The agent that wins the arbitration and obtains access rights to the bus begins the acquisition phase; that agent becomes the bus owner. This agent begins its transfer cycle and holds the arbitration logic in the resolution phase (resolving for the next access rights) until the transfer cycle is completed.

Transfer Cycle

Starting the transfer cycle is the request phase. In this phase, the bus owner (requesting agent) places address and command information on the bus. This information defines the replying agent(s), the type of operation, and the type of address space. The request phase lasts one bus clock cycle.

The reply phase starts immediately after the request phase, in which the replying agent(s) satisfy the request. During this phase, the requesting and replying agents engage in a handshake that synchronizes the data transfer sequence. The reply phase can contain one or more data cycles. The final data transfer is signaled by the requesting agent. During this final transfer, the requesting agent releases ownership of the bus allowing the new bus owner to use the bus immediately. Note how the transfer cycle overlaps the resolution phase of the arbitration cycle to minimize bus dead time.

Exception Cycle

If an agent detects an error during a transfer cycle, it immediately begins an exception cycle. The exception cycle terminates any arbitration cycles and transfer cycles in progress. The exception cycle starts with the signal phase in which the detecting agent activates one of the bus' error lines. This notifies all agents of the problem causing them to terminate any arbitration or transfer cycles. Next the recovery phase begins. During this phase, all agents idle; this allows the bus a fixed amount of idle-time to stabilize before resuming normal operation.

Signal Groups

Overview

The iPSB bus contains five groups of signals, Figure 4, over which the requesting and replying agents can enact the protocol. An asterisk following the signal name indicates that that particular signal or group of signals are active when at their electrical low.

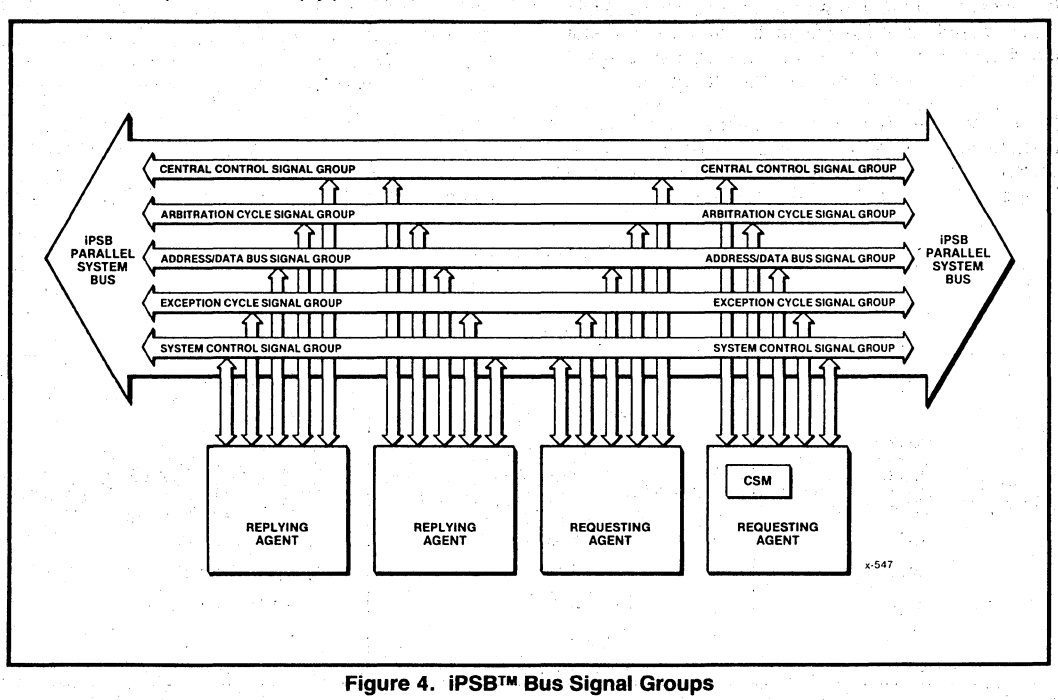


Figure 4. iPSB™ Bus Signal Groups

Arbitration Group

The arbitration signals on the iPSB bus determine which agent gains exclusive access to the bus (which agent is the bus owner). All requesting agents that require access to the bus resources must arbitrate for use of the bus. On being granted bus ownership, an agent begins using the address/data lines to perform a transfer cycle. There are seven signals in the arbitration group: BREQ* and ARB5* through ARB0*.

BREQ* (Bus Request) is an OR-tied signal which is bussed on the backplane. All agents that require access to the bus assert the BREQ* signal.

A particular agent's arbitration ID number is coded on lines **ARB4* through ARB0*** (Arbitration). An agent requiring use of the iPSB bus asserts BREQ* and drives its arbitration ID onto the OR-tied ARB lines. The ARB5* line selects one of two arbitration algorithms: fairness or high priority.

Address/Data Bus Group

This signal group contains the lines used to transfer the address and data information plus their respective byte parity lines. The **AD31* through AD0*** (Address/Data) lines are multiplexed and serve a dual purpose depending upon the phase of the transfer cycle.

During the request phase, they contain the address for the ensuing transfer. This address refers to the byte location for memory and I/O spaces, a processing agent module in message space, and a board slot location in interconnect space. The requesting agent drives these lines during the request phase.

During the reply phase, they contain either eight, sixteen, twenty-four, or thirty-two bits of data. They are driven by the requesting agent for write transfers and by the replying agent for read transfers.

The **PAR3* through PAR0*** (Parity) lines are the byte parity lines associated with the respective bytes of the AD lines. They form even parity with their respective address/data byte.

System Control Signal

The transfer signal group consists of ten signals, SC9* through SC0* (System Control). Agents use these signals to define commands or to report status, depending on the phase of the transfer cycle.

During the request phase, the requesting agent drives SC9* through SC0*. The SC lines provide command information to the replying agent(s). During the reply phase, the requesting agent drives SC9* and SC3* through SC0* with its handshake and additional control information. The replying agent drives the remainder with its handshake and status. Table 1 lists the request and reply phase functions for this group.

Exception Signal Group

The iPSB bus provides a group of two signals for passing indications of exception errors to all agents: **BUSERR*** (Bus Error), and **TIMOUT*** (Time-out).

An agent activates BUSERR* to indicate its detection of a data integrity problem during a transfer. Parity errors on the AD or SC lines and the detection of uncorrectable errors in the memory subsystem are typical of errors signaled on BUSERR. Any agent detecting such errors must signal BUSERR* and all agents must receive BUSERR*.

TIMOUT* is signaled by the CSM whenever it detects the failure of a module to complete a handshake. TIMOUT* is received by all agents on the bus.

Central Control Group

The system control group provides status concerning the operating state of the entire iPSB bus environment.

Table 1. System Control Definition

Signal	Function	
	Request Phase	Reply Phase
SC0	Request Phase	Request Phase
SC1	Lock	Lock
SC2	Data Width 0	End-of-Cycle
SC3	Data Width 1	Requesting Agent Ready
SC4	Address Space 0	Replying Agent Ready
SC5	Address Space 1	Agent Error 0
SC6	Read/Write	Agent Error 1
SC7	Reserved	Agent Error 2
SC8	Parity (SC7-4)	Parity (SC7-4)
SC9	Parity (SC3-0)	Parity (SC3-0)

It consists of seven signals plus the power and ground lines.

The **RST*** (Reset) signal is a system-level initialization signal sent to all agents by the CSM.

The **RSTNC*** (Reset Not Complete) signal is an OR-tied line driven by any agent whose internal initialization sequence is longer than that provided by the RST* signal itself. Due to its OR-tying, RSTNC* remains active until every agent has completed its initialization sequence. Agents cannot perform bus transfer cycles until RSTNC* is inactive.

The CSM provides a **DCLOW** (DC Power Low) signal to all agents as a warning of an imminent loss of DC power. DCLOW is typically generated from a signal supplied by the system power supply on the loss of AC power. Any agent needing to preserve state information in battery backed-up resources should do so upon receiving an active DCLOW.

Accompanying DCLOW for power-down sequencing is the **PROT*** (Protect) signal. The CSM drives PROT* active a short time after it activates DCLOW to inform all bus interfaces to ignore any transitions on the bus as power is lost.

The **BCLK*** (Bus Clock) and **CCLK*** (Constant Clock) signals are supplied by the CSM to all agents. Agents use the BCLK to drive the arbitration and timing state machines on the iPSB bus. The active going edge of BCLK* provides all system timing references. The CCLK* is an auxiliary clock at twice the frequency of BCLK.

An agent uses its **LACHn*** (ID Latch) signal to save the slot ID it receives from the CSM at RST time via the ARB4* through ARB0* lines. The ID latch signal is called LACHn* where the "n" is the card slot to which the ID is assigned. At each card slot, the LACHn* signal is connected to the AD line of the same number. As an example, card slot 7 has a LACH7* signal that is connected to AD7*.

When RST is active, the CSM sends successive slot ID's (0 through 19) on the ARB4* through ARB0* lines while activating the corresponding AD line. Agents know when the ARB lines contain the correct slot number when they see their LACHn line go active.

Power

System power supplied in the iPSB connector includes +5 volts, +12 volts, −12 volts, and facilities for +5 volt battery back-up. Also defined are numerous ground lines some of which are interlaced throughout the connector. The iPSB bus power arrangement conforms to the proposed IEC standard for power railing in IEC 603-2 connectors.

iPSB Bus Protocol

Overview

In the MULTIBUS II specification, both timing diagrams and state-flow diagrams describe the iPSB bus protocol. The state-flow diagrams present the lowest-level and most rigorous definition while the timing diagrams help conceptual understanding. For the purposes of this data book, only the timing diagram description is used.

Arbitration Cycle

An agent that wishes to transfer data on the iPSB bus must begin by performing an arbitration cycle. The cycle performs two functions: first, it gives all agents the opportunity to be granted access to the bus, and second, it eliminates the possibility of more than one agent trying to transfer data on the bus at any one instant. In the case where more than one agent requests access to the bus at the same instant, the arbitration cycle grants access to the agents based upon one of two arbitration algorithms: normal or high priority.

Normal priority mode provides "fairness" or "no starvation", which each agent has an equal opportunity to grant access to the bus. For example, assume all agents request the bus at the same instant. In the normal priority mode, each agent is granted the bus, one by one, until all requests have been serviced. If an already serviced agent desires to use the bus again before all of the original agents are serviced, it will wait until all of original requesting agents have their requests granted. This "round-robin" granting of access ensures that any agent requesting the bus will eventually get it.

The high priority mode allows an agent with high priority to force its way into the arbitration and grant the bus before agents with lesser priority. This means that a high priority agent gets access to the bus quickly, however it can also consume much of the bus that agents with lesser priority never gain access; they will "starve".

At reset, the CSM supplies each agent with its slot ID and its arbitration ID. An agent making a normal priority request activates BREQ*, holds ARB5* inactive, and drives its arbitration ID onto ARB4* through ARB0*. If the ARB lines hold its ID after a specified time (3 bus clocks), this agent won the arbitration and can use the bus once any ongoing transfer completes. However, if the ARB lines do not match its ID (after all, other agents might be also requesting the bus and driving the ARB lines), another agent won the arbitration. The losing agent removes its ID and waits for the next resolution phase before trying again.

An agent makes a high priority request by activating BREQ*, holding ARB5* active (ARB5* selects the arbitration mode), and driving its arbitration ID onto the ARB lines. The high priority algorithm requires that

when a high priority request enters during an arbitration cycle, the request immediately enters the next resolution phase rather than waiting for the next arbitration cycle as do normal priority requests. ARB5* being active causes the other requesting agents to remove their requests guaranteeing the high priority agent access to the bus before any simultaneous normal priority requests. When more than one agent simultaneously makes a high priority request, the agent with the higher priority (lower numerical value) arbitration ID will go first. Figure 5 illustrates the logic required to implement the iPSB bus arbitration. With either priority mode, once an agent owns the bus, it can perform any number of transfer cycles until forced off by arbitration. This characteristic of the arbitration algorithms is called "bus parking".

Transfer Cycle

Transfer cycles consist of two phases: request and reply. For illustration, an example of a access read cycle is shown in Figure 6. During the request phase, the bus owner (requesting agent) uses the transfer cycle signal group (SC lines) to notify the replying agent of the address space (memory, I/O, interconnect, or message), the data width (8-, 16-, 24- or 32-bit), and whether the cycle is read or write. The AD lines contain the desired address for the selected address space. Replying agents know the SC lines contain this request information by the requesting agent activating SC0* (Request Phase). The request phase lasts one clock cycle. All potential replying agents use the request phase to determine whether they contain the addressed resource.

The reply phase starts immediately following the request phase. During this phase the agent with the addressed resource (replying agent) and the requesting agent exchange data and status. Both the requesting and replying agent must agree that the data on the AD lines and the status on the appropriate SC lines are valid via the RQRDY (Requesting agent ready — SC3*)

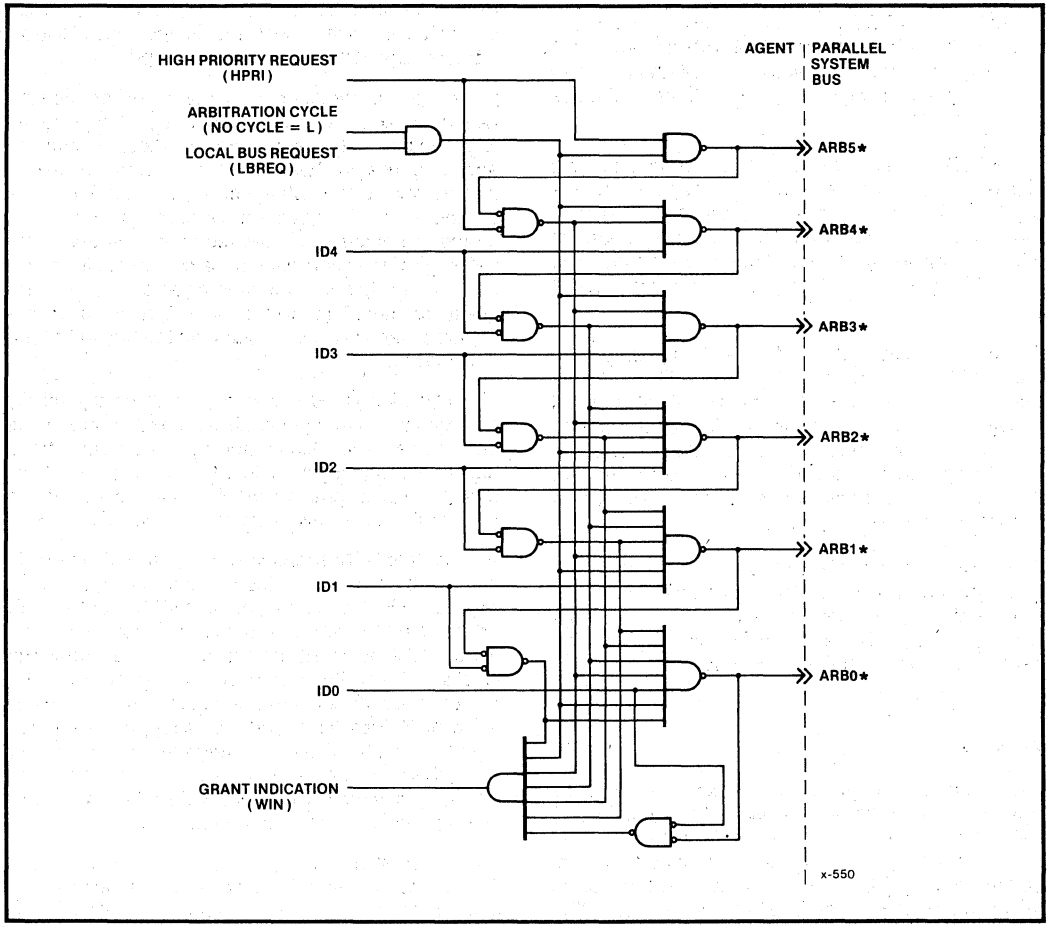


Figure 5. iPSB Bus Arbitration Cycle

and RPRDY (Replying agent ready) handshake lines. Either agent can hold off the transfer by deactivating its ready line. This handshaking supports any speed requesting or replying agent.

The transfer cycle is complete when the requesting agent signals the last data transfer via the End-Of-Cycle (EOC — SC2*). The last bus clock cycle of the transfer is when EOC, RQRDY, and RPRDY are all active simultaneously.

The replying agent has the opportunity to tell the requesting agent if it does not support the requested operation via the agent error (SC5*, SC6*, and SC7*) lines. These lines encode four types of errors: width violation, continuation error, illegal operation, and negative acknowledgment of a message. Trying to extract 32-bits of data from an 8-bit peripheral is an example of a data width violation. Continuation errors occur when attempting sequential access from an agent which does not support them or running off the ending address of a memory board. Writing to a read-only memory is an example of an illegal operation. A replying agent signals a negative acknowledgment to a message transfer cycle if its destination queue is full (the source must perform source queuing). The transfer cycle is terminated by the requesting agent when it detects that the replier is signalling an agent error. If the bus interface is intelligent, it might retry the operation with a different type that the replying agent can support. Other aspects of transfer cycle include the ability of a requesting agent to LOCK the bus via the SC1* line. SC1* is a non-multiplexed signal which inhibits alternate ports of any multi-ported resource being addressed. By locking the bus, the requesting agent guarantees itself exclusive access to a multi-ported bus resource and retains bus ownership for more than one transfer cycle.

As noted in the figure, in addition to parity protection on the address/data lines, the SC lines are also protected by parity. The requesting agent is responsible for the SC parity bits (SC8* and SC9*) during the request phase (it drives all SC lines). The reply phase requires two parity bits: one for those lines driven by the requesting agent and one for those driven by the replier. This ensures all aspects of the transfer cycle have parity protection.

Exception Cycle

The exception cycle is an error reporting mechanism. An agent or the CSM initiates an exception cycle as a result of sensing an exception. If no exception occurs, no exception cycles occur.

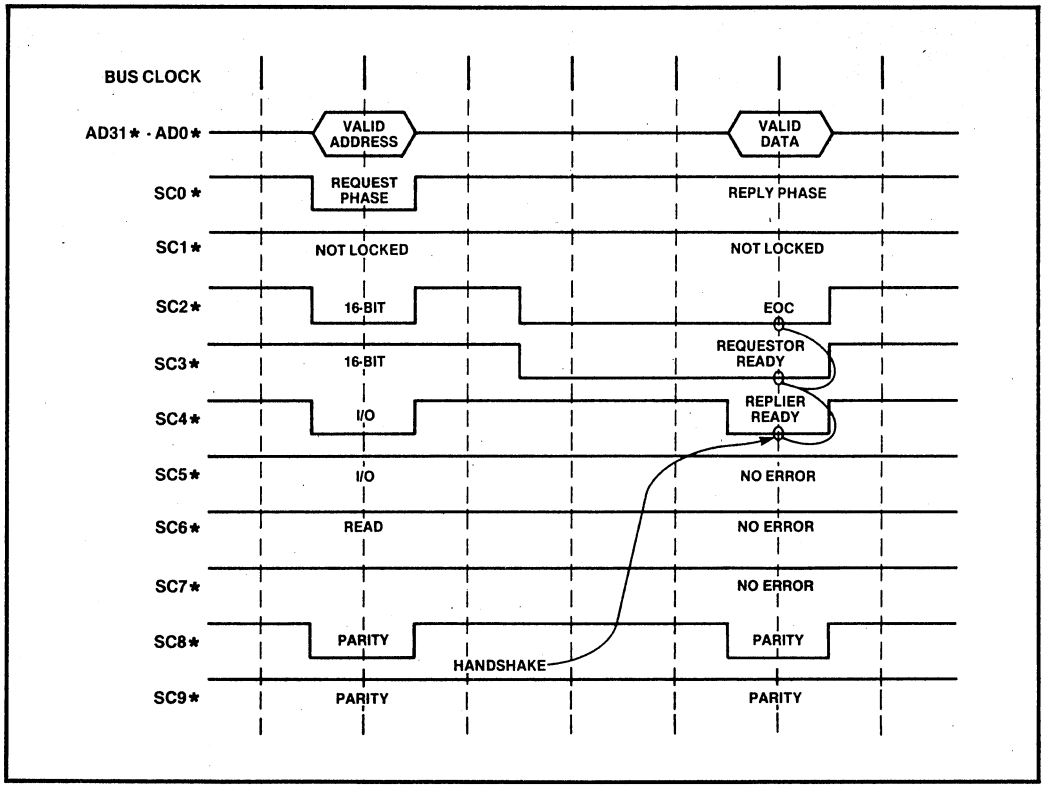


Figure 6. Transfer Cycle Example

The exception cycle has two purposes in the protocol: first, it provides systematic termination of activity on the iPSB bus and second, it provides a stabilization time before allowing agents to resume operation. These two purposes correspond directly to the two phases of the exception cycle: the signal and recovery phases.

The signal phase begins when an agent or a module senses an exception and activates one of the bus error lines. On receiving a bus error, all agents terminate any transfer or arbitration cycles in progress. The net effect of the signal phase is to terminate all bus activity. The signal phase continues until the error-detecting module deactivates the bus error line.

The recovery phase begins after the bus error line becomes inactive. The recovery phase is a fixed-duration delay (in terms of bus clock cycles) that allows time for the iPSB bus signals to settle before starting more transfer cycles.

There are two types of bus exceptions supported by the iPSB bus: timeout and bus error. The CSM monitors the bus to ensure that all data handshakes complete. If for some reason the handshake hangs and exceeds a maximum time limit, the CSM activates the TIMOUT* (Time Out) bus exception line to begin the exception cycle.

An agent sends a bus error exception whenever it determines that the information on the address/data (AD) or the transfer control (SC) lines is in error. Typical examples of bus error exceptions are parity errors and non-recoverable memory errors sensed during a data transfer. Once an error is detected, the agent activates the BUSERR* (Bus Error) signal line to begin the exception cycle.

Mechanical

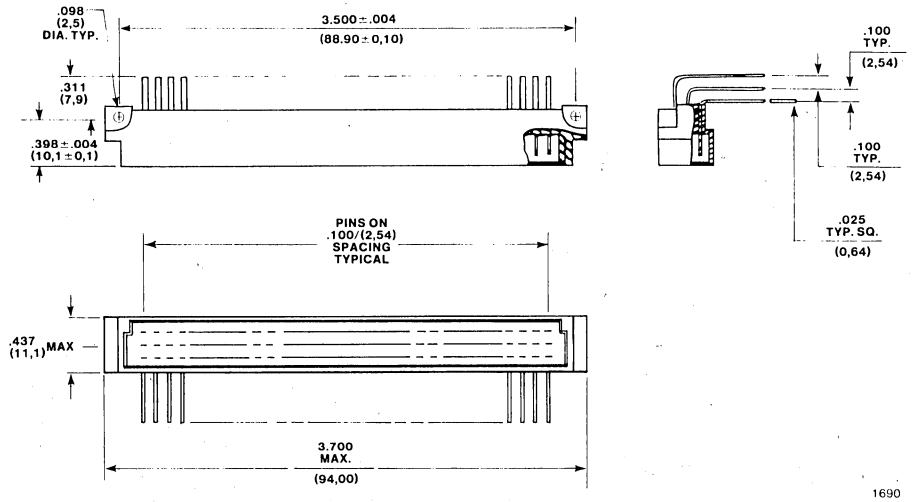
The MULTIBUS II boards, board accessories, and backplanes conform to mechanical standards defined by the International Electromechanical Commission (IEC); these standards are commonly referred to as the Eurocard mechanical standards. This mechanical system offers modular board sizes as defined in standard IEC-297-3 and reliable two-piece connectors as defined in IEC-603-2.

Form Factor

The MULTIBUS II specification calls out two modular board form factors: 233 x 220mm and 100 x 220mm (see Figure 7). The iPSB bus and iLBX II bus portions of the MULTIBUS II system architecture are always defined on the P1 and P2 connectors respectively. However, the user can optionally define the use of the P2 connector if the iLBX II bus is not supported. (The iSSB bus is additionally defined on the P1 connector.)

Connector

MULTIBUS II boards and backplanes use two-piece, 96-pin connectors for both the iPSB bus and the iLBX II bus. The right-angle connectors on the printed board are IEC standard 603-2-IEC-C096-M; the receptacle connectors on the backplane are IEC standard 6-03-2-IEC-C096-F (Figure 8). This connector family is noted for its reliability, availability, and low cost.



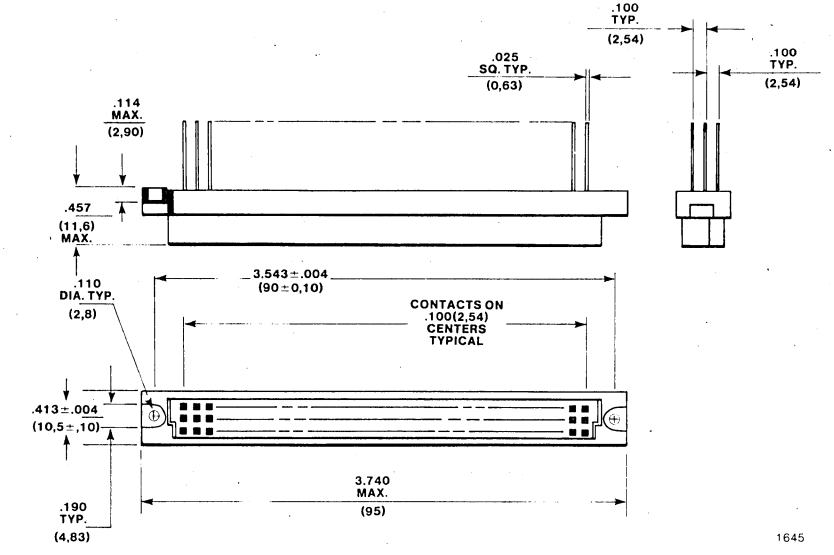


Figure 8. MULTIBUS® II Connectors

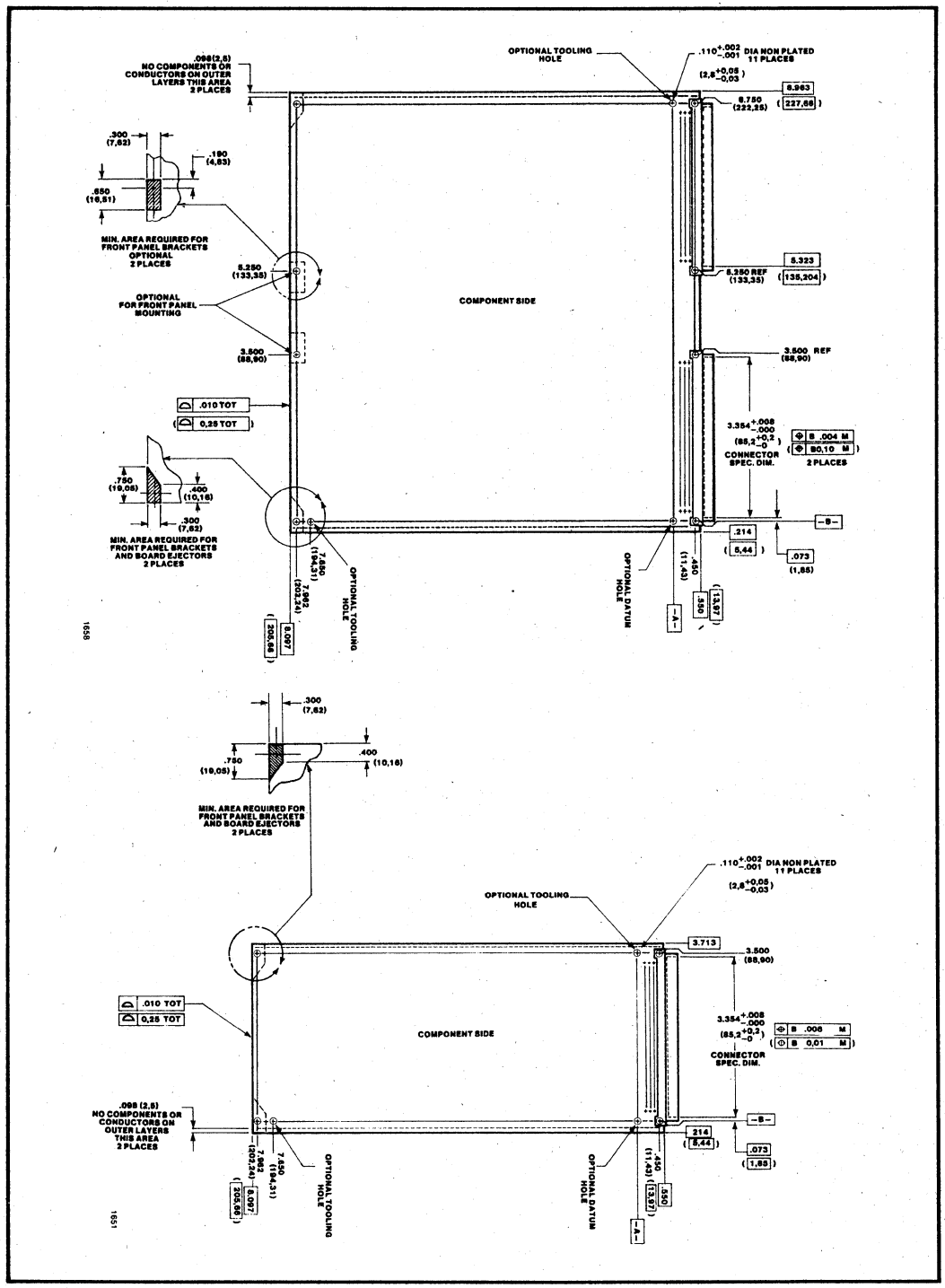


Figure 7. MULTIBUS® II Board Sizes

The pin assignment for the iPSB bus on P1 is shown in Table 2.

Please refer to Intel's MULTIBUS II Bus Architecture Specification Handbook for more detailed information.

Table 2. iPSB Bus Pin Assignments

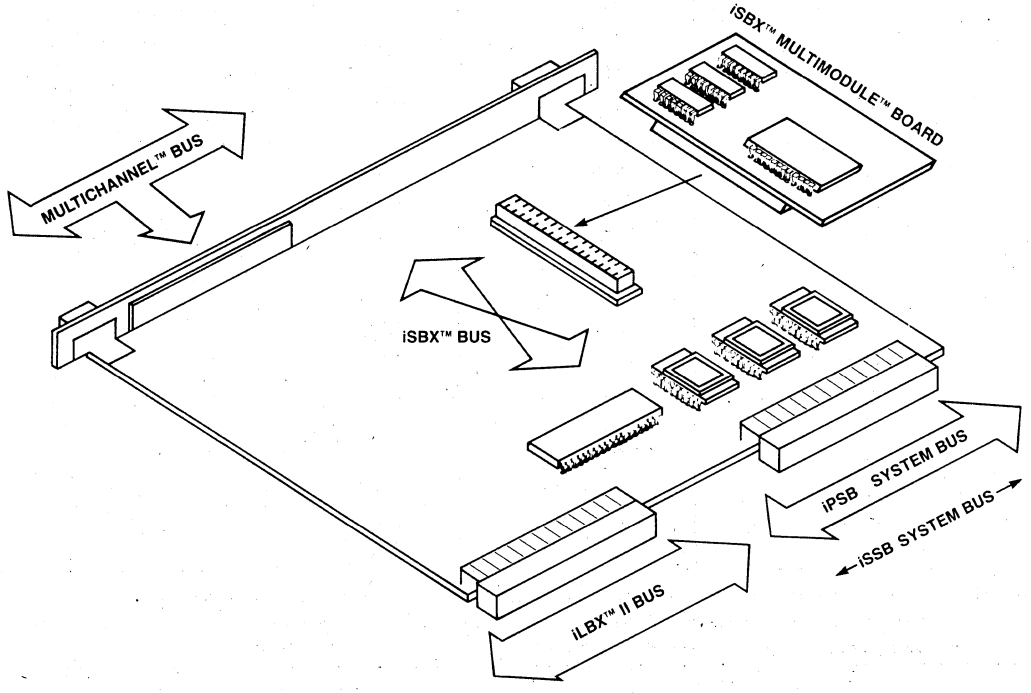
Connector Pin Number	Row A	Row B	Row C
1	0 Volts	PROT*	0 Volts
2	+5 Volts	DCLOW*	+5 Volts
3	+12 Volts	+5 Battery	+12 Volts
4	(Note 2)	SDA (Note 3)	BCLK*
5	TIMOUT*	SDB (Note 3)	0 Volts
6	(Note 1) LACHn	0 Volts	CCLK*
7	AD0*	AD1*	0 Volts
8	AD2*	0 Volts	AD3*
9	AD4*	AD5*	AD6*
10	AD7*	+5 Volts	PAR0*
11	AD8*	AD9*	AD10*
12	AD11*	+5 Volts	AD12*
13	AD13*	AD14*	AD15*
14	PAR1*	0 Volts	AD16*
15	AD17*	AD18*	AD19*
16	AD20*	0 Volts	AD21*
17	AD22*	AD23*	PAR02*
18	AD24*	0 Volts	AD25*
19	AD26*	AD27*	AD28*
20	AD29*	0 Volts	AD30*
21	AD31*	Reserved	PAR3*
22	+5 Volts	+5 Volts	Reserved
23	BUSREQ*	RST*	BUSERR*
24	ARB5*	+5 Volts	ARB4*
25	ARB3*	RSTNC*	ARB2*
26	ARB1*	0 Volts	ARB0*
27	SC9*	SC8*	SC7*
28	SC6*	0 Volts	SC5*
29	SC4*	SC3*	SC2*
30	−12 Volts	+5 Battery	−12 Volts
31	+5 Volts	SC1*	+5 Volts
32	0 Volts	SC0*	0 Volts

NOTES:
1. LACHn* for all agents but the one driving CCLK*; line contains a second CCLK* signal in systems that have more than 12 cardslots.
2. 0 Volts for all agents but the one driving BCLK*; line contains a second BCLK* signal in systems that have more than 12 cardslots.
3. Signal lines SDA and SDB are reserved for the Serial System Bus.

MULTIBUS® II
iSSB Serial System Bus

- **Logical equivalent to the iPSB bus message space**
- **2 megabits/sec serial data rate**
- **Multi-master capability up to 32 nodes**
- **Physical distribution up to 10 meters**
- **Deterministic access protocol**
- **Based upon CSMA/CD (Carrier Sense Multiple Access with Collision Detection)**

The iSSB Serial System Bus is a simple, low cost alternative to the iPSB Parallel System Bus message address space. The message passing interface is identical for both buses; this allows easy migration from one bus to the other with no software changes. The iSSB bus serves as a low-cost replacement for the iPSB bus in applications where cost reduction is required and serves as a complement to the iPSB bus where an alternative bus path is needed for interface control, diagnostics, or redundancy changes.



MULTIBUS® II Physical Diagram

FUNCTIONAL DESCRIPTION

Architectural Overview

The trend toward a more functional VLSI has driven the cost-functionality vector to allow system designers to pack more and more functionality on a given size board while maintaining approximately constant cost. The iSSB Serial System Bus lets VLSI drive the cost-functionality vector in the other direction; dramatically reduce the cost while maintaining roughly constant functionality. It accomplishes this by reducing the **interconnect cost** and allowing **physical distribution** of modules.

Reduced Interconnect Cost

Most systems today use a parallel interface to interconnect boards within the system. Frequently the cost to provide this interconnect is a significant percentage of the total system cost. Connectors, backplanes and interface logic are all part of this interconnect cost.

The iSSB Serial System Bus dramatically reduces the interconnect cost by replacing the parallel interface's multiple-line connector and backplane with a simple twisted-pair interface using telephone-type connectors. It also reduces the interface logic to a single VLSI component as opposed to the multiple components required in a parallel interface.

Physical Distribution

Being tied to a backplane or bulky ribbon cable limits the system designer's mechanical flexibility in constructing a system from multiple modules. The iSSB bus frees him of these restrictions by letting him physically distribute the system modules up to 10 meters apart.

Structural Features

Physical Characteristics

The iSSB bus consists of a maximum of 32 nodes which can be distributed over a maximum of 10 meters of cable. The nodes may be distributed along an external cable segment or clustered into backplanes as shown in Figure 2. Each backplane may contain up to 20 nodes, the maximum number of cardslots in a iPSB bus backplane.

Clustered systems use repeaters as a connection between backplanes and the iSSB bus cable. The repeaters isolate the cable from the excessive capacitive load on the backplane.

Access Protocol

The iSSB bus employs an access protocol called Carrier-Sense-Multiple-Access with collision detection (CSMA/CD). The CSMA/CD protocol allows agents to transmit data whenever they are ready.

In CSMA/CD operation, an agent with data to transmit looks at the iSSB bus for traffic before beginning a transmission. If the bus is not idle, the agents waits until the line becomes idle and until an interframe space has passed. After both events, the agent begins transmission of the message.

It is possible for more than one agent to initiate a transmission at the same time; in that case, a collision occurs on the bus. The protocol handles collisions on the iSSB bus via a deterministic collision resolution algorithm that uses time slotting.

The deterministic collision algorithm guarantees a time slot during which each agent can transmit without interference from other agents. The resolution guaran-

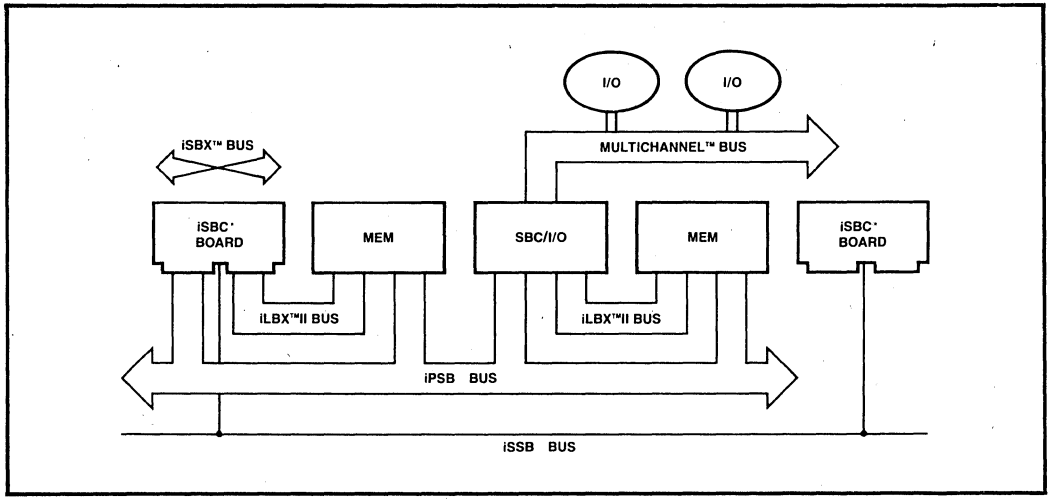


Figure 1. MULTIBUS® II Bus Architecture

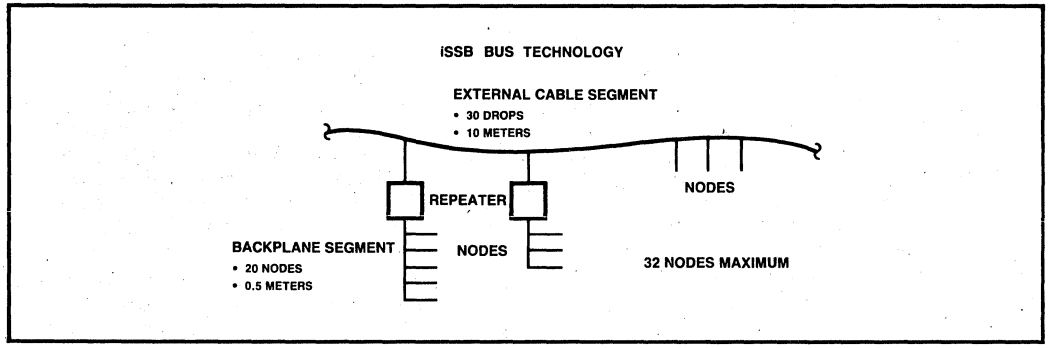


Figure 2. Typical iSSB Bus System Configurations

tees fair access to all agents. This type of collision resolution provides a real-time response that allows agents to resolve collisions in a finite time period.

Error Control

The iSSB bus uses a 16-bit CRC (Cyclic Redundancy Check) in order to provide error detection. Used in conjunction with an intelligent interface, this allows the iSSB to look as reliable as the iPSB bus even though it is up to 10 meters long.

Physical Interface

The physical iSSB bus interface consists of two signal lines (the SDA and SDB lines) that are included as part of the iPSB bus backplane design and may be extended via a 2-wire cable that connects to a repeater, typically located on the CSM. Agents encode data on the complementary, open-collector signal lines as shown in Table 1.

Table 1. iSSB Bus Signal Line Encoding

SDA Line	SDB Line	Line Condition
0	0	collision
0	1	logic 0
1	0	logic 1
1	1	idle

The portion of the signal lines within the backplane is designed to operate in a high-noise environment such as a heavily loaded backplane. Cable extensions to the iSSB bus must adhere to normal transmission line requirements.

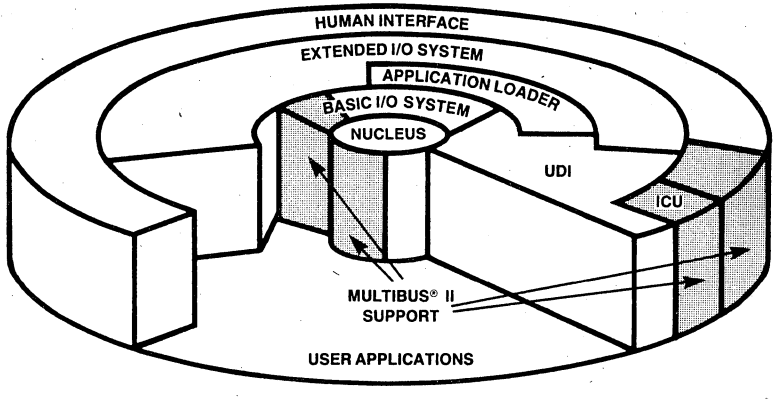
To further improve reliability, the bus interface includes receivers that sample the data and filter out noise which may be coupled from the surrounding environment.

Please refer to Intel's MULTIBUS® II Bus Architecture Specification Handbook for more detailed information.

iRMX™ 86-MULTIBUS® II SUPPORT PACKAGE

- **MULTIBUS® II support for iSBC® 286/100 applications in Real Address Mode, including support for the SCSI peripheral interface and up to 1 megabyte addressability**
- **Functions in conjunction with the iRMX™ 86 Release 6 Operating System**
- **Interprocessor Signal Support**
- **Automatic software configuration of memory boards**
- **Support for battery backed-up, global time-of-day clock**
- **Extendable to allow addition of custom device drivers**

The iRMX™ 86-MULTIBUS® II Support Package, functioning with the iRMX 86 Release 6 Operating System software, provides the ability to execute all configurable layers of the iRMX 86 software in the MULTIBUS II environment (iRMX 86-MULTIBUS II Operating System). Applications in Real Address Mode are supported for the iSBC® 286/100 board, including support for the SCSI peripheral interface and all iSBX™ boards supported by iRMX 86 Release 6, as well as support for iAPX 286 component applications.



NEW IN iRMX™ 86 MULTIBUS® II OPERATING SYSTEM

iRMX™ VLSI Operating System

SEPTEMBER, 1984
ORDER NUMBER: 280057-001

FUNCTIONAL DESCRIPTION

Overview

The iRMX 86 MULTIBUS II package contains system modules that replace portions of the iRMX 86 Release 6 Operating System, allowing the iRMX 86 Operating System to execute in a MULTIBUS II environment. All the functions available in the IRMX 86 Operating System are available in the iRMX 86-MULTIBUS II Operating System. For a complete description of these functions, their value, and performance, please refer to the Release 6 iRMX 86 Operating System Data Sheet (order number 210885-002).

This functional description section describes the new features provided by the iRMX 86 MULTIBUS II package. These new features add the new capabilities required for OEMs to execute the iRMX 86 Operating System in a MULTIBUS II environment for iSBC 286/100 or iAPX 286 applications in Real Address Mode.

Interprocessor Signal Support

In a MULTIBUS II system, interprocessor communication and synchronization is done via messages over the bus. This communication includes data-less messages to signal that an event has occured. The iRMX 86 MULTIBUS II package supports signal messages using the Message Interrupt Controller (MIC) Component. The major advantage of signal message support is the ability for a host cpu board to send or receive signal messages from up to 254 distinct sources, with the priorities of each message being based on the sending or receiving task's priority. Signal messages are not tied to hardware interrupt levels and priorities as external interrupts were in the MULTIBUS I environment.

Automatic Software Configuration of Memory Boards

The iRMX 86-MULTIBUS II Operating System has the option of automatically configuring memory boards. The addresses for each board are defined sequentially in relation to the physical placement of each board in the card cage. This feature allows for the swapping, adding, and deleting of memory boards in the system on a dynamic basis.

Accurate Time-of-Day Clock Support

Resident in every MULTIBUS II system is a Central Services Module (iSBC CSM/001 board). The CSM board contains a battery backed-up, global time-of-day clock. The iRMX 86-MULTIBUS II Operating System uses this clock to automatically initialize the time-of-day clock maintained by the operating system.

Custom Device Driver Support

Like the iRMX 86 Operating System, the iRMX 86-MULTIBUS II Operating System is extendable to support user value-added custom device drivers. This feature allows the system to be more closely tailored to meet a specific application requirement and expands the list of supported hardware products. The user need not purchase source code to write a custom driver and can configure the driver into the system at configuration time. Custom drivers can use the Message Interrupt Controller (MIC) to pass signal messages.

SPECIFICATIONS

Below is the list of supported products for the iRMX 86 MULTIBUS II Support Package.

Supported Software Products

iRMX 86 Release 6 Operating System

Supported Hardware Products

Components:

iAPX 286 Microprocessor (Real Address Mode only)
80287 Numeric Data Processor Extension
8253 and 8254 Programmable Interval Timers
8259A Programmable Interrupt Controller (PIC)
8255 Programmable Parallel Interface (PPI)
82530 Serial Communications Controller (SCC)
82258 Advanced DMA Controller (ADMA)
Bus Arbiter Controller (BAC)
Message Interrupt Controller (MIC)

iSBC® MULTIBUS® II Board Products:

iSBC 286/100 Single Board Computer (Real Address Mode only)
iSBC CSM/001 Central Services Module
iSBC MEM/312, 310, 320, 340 cache-based memory
iSBX 218(A) Flexible Diskette Multi-Module Controller
iSBX 251 Bubble Memory Multi-Module
iSBX 270 CRT Light Pen and Keyboard Interface
iSBX 350 Parallel Port (Centronics-type Printer Interface)
iSBX 351 Serial Communications Port

AVAILABLE LITERATURE

iRMX 86-MULTIBUS II Support Package Reference Manual (order number 147127)

There are four manual kits supplied with the iRMX 86 Release 6 Operating System and are available under the order numbers shown in the iRMX 86 Operating System Data Sheet (order number 210885-002)

ORDERING INFORMATION

The iRMX 86 MULTIBUS II Package is available under a number of different licensing options. Obtaining a license for the iRMX 86 Release 6 Operating System is a pre-requisite to licensing the iRMX 86 MULTIBUS II Package. Reconfigurable object libraries are provided on: 1) Double-density single-sided ISIS-formatted 8″ diskettes; 2) Double-density, single-sided iRMX 86-formatted 8″ diskettes; 3) Double-density, double-sided, iRMX 86-formatted 5.25″ diskettes. ISIS-format diskettes may be used on Series III Development Systems. The iRMX 86-format may be used on Series IV Development Systems (5.25″ diskettes) or any iRMX 86-based system supporting the appropriate disk drivers, compilers and development environment.

The OEM license options listed here allow users to incorporate the iRMX 86 MULTIBUS II package into their applications. Each use requires payment of an Incorporation Fee.

ORDER CODE	DESCRIPTION
iRMX 86 II BRO:	Double-density, single-sided 8″ ISIS-format OEM license.
iRMX 86 II ERO:	Double-density, single-sided 8″ iRMX 86-format OEM license.
iRMX 86 II JRO:	Double-density, double-sided 5.25″ iRMX 86-format OEM license.
iRMX 86 II KIT BRO:	Includes iRMX 86 Release 6. Double-density, single-sided 8″ ISIS format OEM license.
iRMX 86 II KIT ERO:	Includes iRMX 86 Release 6. Double-density, single-sided 8″ iRMX 86-format OEM license.
iRMX 86 II KIT JRO:	Includes iRMX 86 Release 6. Double-density, double-sided 5.25″ iRMX 86-format OEM license.

Other licensing options include prepayment of all future incorporation fees and single use rights for a single machine.

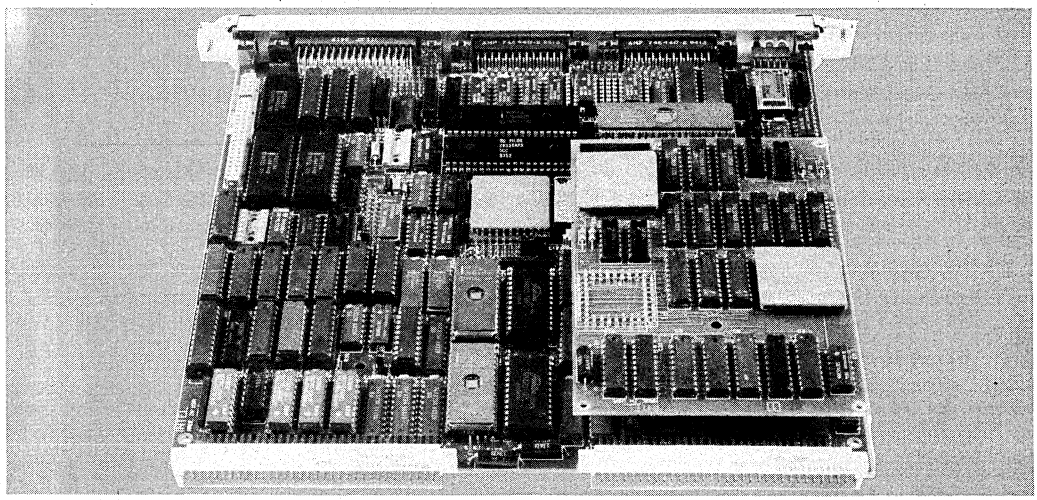
Each option includes 90 days of support service that provides Software Problem Report Service and copies of System Updates that occur during this period.

As with all Intel software, purchase of any of these options requires the execution of a standard Intel Master Software License. The specific rights granted to users depend on the specific option and the license signed.

iSBC® 286/100 MULTIBUS® II SINGLE BOARD COMPUTER

- 8 MHz iAPX 286 Microprocessor
- MULTIBUS® II iPSB (Parallel System Bus) interface for multimaster configurations and multiprocessing system expansion
- MULTIBUS® II iLBX™ II (Local Bus Extension) interface for improved high-speed memory expansion
- MULTIBUS® II interconnect space for software configurability and diagnostics
- Resident firmware to support Built-In-Self-Test (BIST) power-up diagnostics
- Optional 80287 Numeric Data Co-Processor (socket on-board)
- iSBX™ bus interface connector for I/O expansion bus
- Four DMA channels supplied by the 82258 Advanced DMA controller with 8 MBytes/sec. transfer rate
- 16 levels of vectored interrupt control. Up to 255 distinct interrupt sources and 255 interrupt destinations are supported using message-based interrupts
- Two 28-pin JEDEC sites
- 24 programmable I/O lines configurable as SCSI interface, Centronics interface or general purpose I/O
- Two programmable serial interfaces, one RS 232C, the other RS 232C or RS 422A compatible
- Double-high standard Eurocard form factor

The iSBC® 286/100 Single Board Computer takes advantage of the MULTIBUS® II System Architecture for OEM applications. The combination of the iAPX 286 CPU and two new bus structures, the MULTIBUS II Parallel System Bus (iPSB bus) and the Local Bus Extension II (iLBX™ II bus) make the iSBC 286/100 board uniquely suited to high performance, multimaster system applications. The iSBC 286/100 board support of the MULTIBUS II interconnect space provides new software configuration ease and access to Built-In-Self-Test (BIST) power-up diagnostics. The board is a complete microcomputer system on a 8.7 x 9.2 inch double-high Eurocard printed circuit board.



SEPTEMBER, 1984
ORDER NUMBER: 280076-001

FUNCTIONAL DESCRIPTION

Overview

The iSBC 286/100 Single Board Computer utilizes the powerful 8 MHz iAPX 286 CPU within the MULTIBUS II system architecture to provide a high performance multiprocessing 16-bit solution. Figure 1 shows a typical MULTIBUS II multiprocessing system configuration. Overall system performance is enhanced by the iLBX II bus which allows 0 wait state high speed memory execution.

Architecture

The iSBC 286/100 board supports the new iPSB bus features of interconnect space, Built-In-Self-Test (BIST) diagnostics, and message based interrupts. These new features are described in the following sections. Besides taking advantage of the MULTIBUS II system architecture, the iSBC 286/100 board has complete single board computer functionality including iSBX™ bus expansion, 80287 co-processor option, advanced DMA control, JEDEC memory sites and expansion, SCSI configurable parallel interface, serial I/O, and programmable timers. Figure 2 shows the iSBC 286/100 board block diagram.

Central Processing Unit

The central processor for the iSBC 286/100 board board is the 80286 CPU operating at an 8.0 MHz clock rate. The 80286 CPU is upwardly compatible with Intel's iAPX 86 and iAPX 186 CPUs. The 80286 CPU

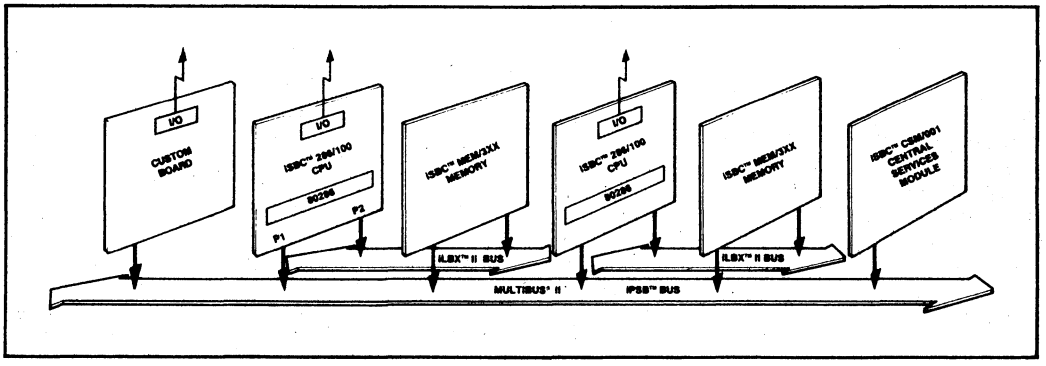


Figure 1. Typical MULTIBUS® II Multiprocessing System Configuration

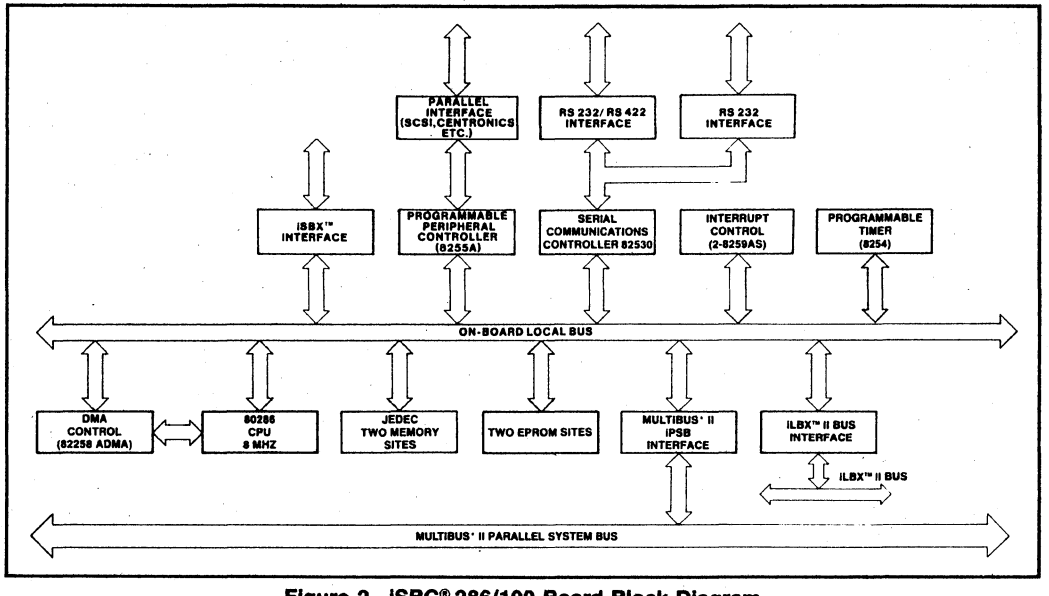


Figure 2. iSBC® 286/100 Board Block Diagram

runs iAPX 86 and 186 code at substantially higher speeds due to a parallel chip architecture. Numeric processing power may be enhanced with the optional 80287 numeric co-processor. The 80286 CPU operates in two modes: iAPX 86 real address mode and protected virtual address mode. In iAPX 86 real address mode, programs use real address with up to one megabyte of address space. In protected virtual address mode, the 80286 CPU automatically maps 1 gigabyte of virtual address per task into a 16 megabyte real address space. This mode also provides the hardware memory protection for the operating system. The operating mode is selected via CPU instructions.

INTERCONNECT SPACE SUPPORT

MULTIBUS II interconnect space is a standardized set of read-only and software configurable registers; the read-only registers hold information such as board type, the software configurable registers allow read and write operations under software control.

The iSBC 286/100 board uses MULTIBUS II interconnect space capabilities for dynamic software system configuration and remote diagnostics and testing. A software monitor, e.g. the SDM 286 software monitor, can be used to dynamically change iLBX II bus memory sizes, disable on-board resources such as PROM or JEDEC sites, read if iSBX bus or PROM are installed as well as access the results of Built-In-Self-Tests or user installed diagnostics.

BUILT-IN-SELF-TEST (BIST) DIAGNOSTICS

Resident firmware to support MULTIBUS II Built-In-Self-Test power-up diagnostics is supplied by the on-board microcontroller. These BISTs improve the reliability and error reporting and recovery capability of MULTIBUS II boards.

These confidence tests and diagnostics not only improve reliability but also reduce manufacturing and maintenance costs for the OEM user. LED 1 (labelled BIST) is a LED on the front panel used to indicate the status of the power up diagnostics. It is turned on when the BIST starts running and is turned off when the BIST completes successfully.

Error Reporting and Recovery

MULTIBUS II Parallel System Bus and iLBX II provide bus transmission and bus parity error detection signals. Error information is logged in the Bus Arbiter Controller and a bus error interrupt generated. Information on the error source for reporting or recovery purposes is available to software through the iSBC 286/100 board interconnect space registers.

INTERRUPT CONTROL

In a MULTIBUS II system, external interrupts (interrupts originating off the CPU board) are messages over the bus rather than signals on individual lines. Message based interrupts are handled by the Message Interrupt Controller component located on the bus interface piggyback. This means that 1 interrupt line can handle interrupts from up to 256 sources.

Two on-board 8259A programmable interrupt controllers are used for processing on-board interrupts. One is used as the master and the other as the slave. Table 1 includes a list of devices and functions supported by interrupts.

iSBX® BUS MULTIMODULE™ ON-BOARD EXPANSION

One iSBX bus MULTIMODULE connector is provided. Through this connector, additional on-board I/O function may be added. iSBX bus MULTIMODULE boards optimally support functions provided by VLSI peripheral components such as additional parallel and serial I/O, analog I/O, and graphics control. The iSBX bus connector on the iSBC 286/100 board provides all signals necessary to interface to the local on-board bus including 16 data lines and DMA for maximum data transfer rates. MULTIMODULE boards designed with 8-bit data paths and using the 8-bit iSBX bus connectors are also supported. A broad range of iSBX bus MULTIMODULE options are available from Intel. Custom iSBX bus modules may also be designed. An iSBX bus interface specification and mating connectors are available from Intel.

MATH PROCESSOR

The 80287 Math Processor can be installed in the iSBC 286/100 board by the user. The 80287 Math Processor is connected to dedicated processor signal lines which are pulled to their inactive state when the 80287 Math Processor is not installed. This enables the user to detect via software that the 80287 socket is occupied. The 80287 Math Processor runs asynchronously to the 80286 clock. The operating frequency of the 80287 is set by the crystal of the 8284A clock generator which is 24MHz.

DMA CONTROL

Four DMA channels are supplied on the iSBC 286/100 board by the 82258. The 82258 is an advanced DMA (Direct Memory Access) Controller design especially for the 16-bit 80286 microprocessor. It has 4 independent DMA channels which can transfer data at rates up to 8 Megabytes/second (8 MHz clock) in a 80286 system. This large bandwidth allows the user to handle very fast data transfer or a large number of concurrent peripherals.

Table 1. Interrupt Devices and Functions

Device	Function	Number of Interrupts
MULTIBUS® II Interface	Message-based interrupt requests from theiPSB bus via 84120 Message Interrupt Controller.	1 interrupt from up to 256 sources
8751 Interconnect Controller	BIST control functions	1
82530 Serial Controller	Transmit buffer empty, receive buffer full and channel errors	1 interrupt from 10 sources
8254 Timers	Timers 0, 1, 2 outputs; function determined by timer mode	3
8255A Parallel I/O	Parallel port control	2
iLBX™ bus interface	Indicates iLBX™ II bus error condition	3
iPSB bus interface	Indicates transmission error on iPSB bus	1
iSBX® bus connector	Function determined by iSBX® bus MULTIMODULE® board	2
Edge sense out	Converts edge triggered interrupt to a level	1
Bus fail-safe timer	Indicates addressed MULTIBUS® II device has to not responded to command within 10 msec	1
Power-fail	External/Power-Fail interrupts	1

MEMORY CAPABILITIES

The local memory of the iSBC 286/100 board consists of two groups of byte-wide sites. The first group of two sites are reserved for EPROM or ROM and are used for the BIST power-up diagnostic firmware. The second group of two sites support JEDEC standard 28-pin devices.

SCSI PERIPHERAL INTERFACE

The iSBC 286/100 board includes a parallel peripheral interface that consists of three 8-bit parallel ports. As shipped, these ports are configured for general purpose I/O. The parallel interface may be reconfigured to be compatible with the SCSI (Small Computer System Interconnect) disk interface by adding user supplied and programmed Programmable Array Logic (PAL) devices, reconfiguring jumpers and installing a user-supplied 74LS460-1 device.This process and PAL codes are detailed in the iSBC 286/100 Board Manual (#146705-001). Alternatively, the parallel interface may be reconfigured as a Centronics compatible line printer interface by adding one PAL and reconfiguring jumpers. Both interfaces may use the 82588 DMA controllers for data transfers.

The SCSI (Small Computer System Interconnect) interface allows multiple mass storage peripherals such as Winchester disk drives, floppy disk drives, and tape drives to be connected directly to the iSBC 286/100 board. A sample SCSI application is shown in Figure 3. The SCSI interface is compatible with SCSI controllers such as: Adaptek 4500, DTC 1410, Iomega Alpha 10, Shugart 1601 and 1610, Vermont Research 8403, and Xebec 1410.

The Centronics interface requires very little software overhead since a PAL device is used to provide necessary handshake timing. Interrupts are generated for printer fault conditions and a DMA request is issued for every character.

SERIAL I/O

The 82530 Serial Communications Controller (SCC) is used to provide 2 channels of serial I/O. The SCC generates all baudrate clocks and provides loopback capability on both channels. Channel B is RS232C only and is configured as a DCE (may be connected directly to a Display Terminal). Channel A is factory-default configured for DCE RS232C operation. Channel A may be reconfigured by the user for DTE or RS422 operation (or both). For RS422 operation, the user must install RS422 drivers and receivers (not supplied). While in RS422 mode, multi-drop connections are supported.

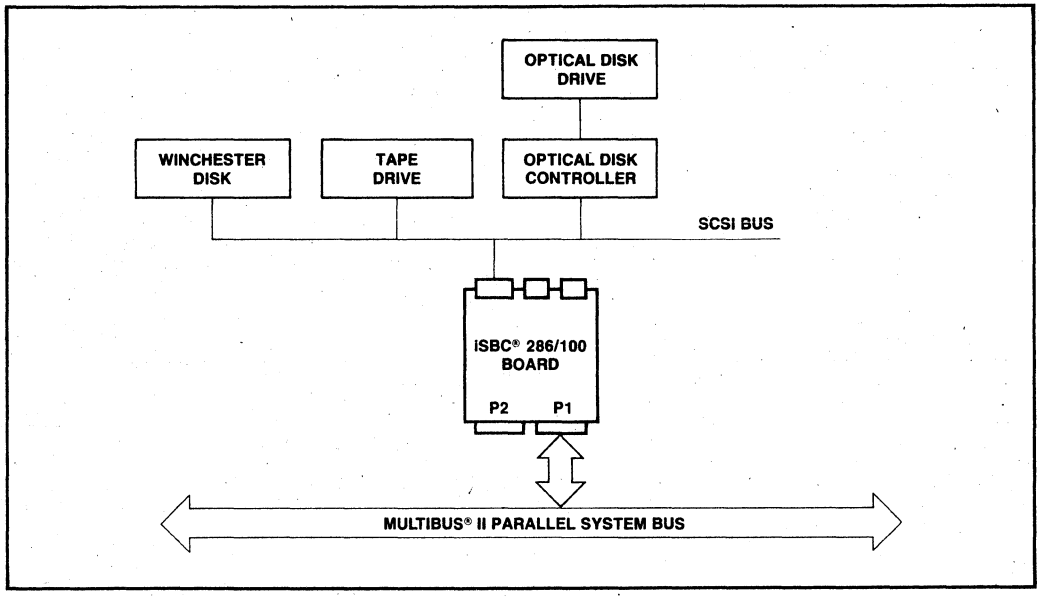


Figure 3. Sample SCSI Applications

The 82258 ADMA can be programmed to support both channels A and B to perform movement of large bit streams or blocks of data.

PROGRAMMABLE TIMERS

The iSBC 286/100 board provides three independent, fully programmable 16-bit interval timers/event counters utilizing the Intel 8254 Programmable Interval Timer. Each counter is capable of operating in either BCD or binary modes. Three of these timers/counters are available to the systems designer to generate accurate time intervals under software control. Routing for the outputs of these counters is jumper selectable. The outputs may be independently routed to the 8259A Programmable Interrupt Controller to count external events or provide baud rate generation. The system software configures each timer independently to select the desired function. Seven functions are available as shown in Table 2. The contents of each counter may be read at any time during system operation.

SOFTWARE SUPPORT

The iRMX™ 86 MULTIBUS II Software Support Package, functioning with the iRMX 86 Release 6 Operating System software, provides the ability to execute all configurable layers of the iRMX 86 software in the MULTIBUS II environment. Applications in Real Address Mode are supported for the iSBC 286/100 board, including support for the SCSI peripheral interface and all iSBX bus boards supported by iRMX 86 Release 6 Operating System, as well as support for iAPX 286 component applications.

The iRMX 86 MULTIBUS II package contains system modules that replace portions of the iRMX 86 Release 6 Operating System, allowing iRMX 86 to execute in a MULTIBUS II environment. All the functions available in the iRMX 86 Operating System are available in the iRMX MULTIBUS II Operating System. For a complete description of these functions, their value, and performance, please refer to the Release 6 iRMX 86 Operating System Data Sheet (order number 210885-002).

Language support for the iSBC 286 boards real address mode includes Intel's ASM 86, PL/M 86, PASCAL and FORTRAN as well as many third party 8086 languages. Language support for virtual address mode operation includes ASM 286, PL/M 286, PASCAL and C. Programs developed in these languages can be down loaded from an Intel Series III or Serivs IV Development System to the iSBC 286 board via the iSDM™ 286 System Debug Monitor Release 2. The iSDM 286 monitor also provides on-target program debugging support including breakpoint and memory examination features.

The MULTIBUS II Interconnect Space Registers allow the software to configure boards eliminating much of the need for jumpers and wire wraps. The iSDM 286 Monitor can initialize these registers at configuration time using user-defined variables. The monitor can also automatically configure memory boards, defining the addresses for each board sequentially in relation to the board's physical placement in the card cage. This feature allows for the swapping, adding, and deleting of memory boards on a dynamic basis.

Table 2. Programmable Timer Functions

Function	Operation
Interrupt on terminal count	When terminal count is reached, an interrupt request is generated. This function is extremely useful for generation of real-time clocks.
Programmable one-shot	Output goes low upon request of an external trigger edge or software command and returns high when terminal count is reached. This function is retriggerable.
Rate generator	Divide by N counter. The output will go low for one input clock cycle, and the period from one low going pulse to the next is N times the input clock period.
Square-wave rate generator	Output will remain high until one-half the count has been completed, and go low for the other half of the count.
Software triggered strobe	Output remains high until software loads count (N). N counts after count is loaded, output goes low for one input clock period.
Hardware triggered strobe	Output goes low for one clock period N counts after rising edge counter trigger input. The counter is retriggerable.
Event counter	On a jumper selectable basis, the clock input becomes an input from the external system. CPU may read the number of events occurring after the counter "window" has been enabled or an interrupt may be generated after N events occur in the system.

SPECIFICATIONS

Word Size

Instruction: 8-, 16-, 24-, 32-, or 40-bits

Data: 8- or 16-bits

System Clock

CPU: 8.0 MHz

Numeric Co-Processor: 8.0 MHz

Cycle Time

Basic Instruction: 8.0 MHz - 375 ns; 250 ns (assumes instruction in queue)

NOTE: Basic instruction cycle is defined as the fastest instruction time (i.e., two clock cycles)

Memory Capacity (Maximum)

EPROM: 2732, 8K bytes; 2764, 16K bytes; 27128, 32K bytes; 27256, 64K bytes; 27512, 128K bytes

E²PROM: 2817A, 4KBytes

IRAM: 2186, 16K bytes

NOTES: Two local sites must contain BIST or user-supplied boot-up EPROM.

I/O Capability

Parallel: SCSI, Centronics, or general purpose I/O

Serial: Two programmable channels using one 82530 Serial Communications Controller

Timers: Three programmable timers using one 8254 Programmable Interrupt Controller

Expansion: One 8/16-bit iSBX MULTIMODULE connector

Interrupt Capacity

Potential Interrupt Sources:
255 individual and 1 broadcast

Interrupt Levels:
16 vectored requests using two 8259As and the 80286s NMI line

Serial Communications Characteristics

Asynchronous Modes:

- 5-8-bit character; odd, even, or parity; 1, 1.5, or 2 stop bits
- Independence transmit and receive clocks, 1X, 16X, 32X, or 64X programmable sampling rate
- Error Detection: Framing, Overrun and Parity
- Break detection and generation

Bit synchronous Modes:

- SDLC/HDLC flag generation and recognition
- Automatic zero bit insertion and detection
- Automatic CRC generation and detection (CRC 16 or CCITT)
- Abort generation and detection
- I-field residue handling
- SDLC loop mode operation
- CCITT X.25 compatible

Byte synchronous Modes:

- Internal or external character synchronization (1 or 2 characters)
- Automatic CRC generation and checking (CRC 16 or CCITT)
- IBM Bisync compatible

Common Baud Rates

Baud Rate	Synchronous (x1 Clock)	Asynchronous (x16 Clock)
	Time Constant	Time Constant
64 K	36	—
48 K	49	—
19.2 K	126	6
9600	254	14
4800	510	30
2400	1022	62
1800	1363	83
1200	2046	126
300	8190	510
110	—	1394

Timers

Input Frequencies:
1.23 MHz ± 0.1% or 4 MHz ± 0.1%
(Jumper Selectable)

Output Frequencies/Timing Intervals

	Single Timer/Counter		Dual Timer/Counter (two timers cascaded)	
	Min.	Max.	Min.	Max.
Real-time interrupt	667 ns	53.3 ms	1.33 ms	58.2 min
Programmable one-shot	667 ns	53.3 ms	1.33 ms	58.2 min
Rate generator	18.8 Hz	1.50 MHz	0.000286 Hz	750 KHz
Square-wave rate generator	18.8 Hz	1.50 MHz	0.000286 Hz	750 KHz
Software triggered strobe	667 ns	53.2 ms	1.33 ms	58.2 min.
Hardware triggered strobe	667 ns	53.3 ms	1.33 ms	58.2 min.
Event counter	—	8.0 MHz	—	—

INTERFACES

iPSB Bus: All signals TTL compatible

iLBX™ II Bus: All signals TTL compatible

iSBX™ Bus: All signals TTL compatible

SERIAL I/O - Channel A:
RS232C/RS422 compatible, configurable as a data set or data terminal;

Channel B:
RS232C compatible, configured as data set

Timer: All signals TTL compatible

Interrupt Requests: All TTL compatible

CONNECTORS

Location	Function	Part #
P1	iPSB bus	603-2-IEC-C096-F
P2	iLBX™ II Bus	603-2-IEC-C096-F

PHYSICAL DIMENSIONS

The iSBC 286/100 board meets all MULTIBUS II mechanical specifications as presented in the MULTIBUS II specification (#146077)

Double-High Eurocard Form Factor:

Depth: 220mm (8.6 in.)

Height: 233mm (9.2 in.)

Front Panel Width: 20mm (.784 in.)

Weight: 33 oz.

ENVIRONMENTAL CHARACTERISTICS

Operating Temperature: 0° to 55° C airflow at 200 linear feet per minute

Operating Humidity: To 90% (without condensation)

ELECTRICAL CHARACTERISTICS

The maximum power required per voltage is shown below. These numbers do not include the power required by the optional memory devices, SCSI PALs, or expansion modules.

Voltage (volts)	Max. Current (amps)	Max. Power (watts)
+5	10.1	53
+12	50 mA	.06
−12	50 mA	.06

REFERENCE MANUALS

iSBC 286/100 Board Manual (#146705-001)

Intel MULTIBUS II Bus Architecture Specification (#146077)

Manual may be ordered from any Sales Representative, Distribution Office, or from the Intel Literature Department, 3065 Bowers Ave., Santa Clara, CA 95051

ORDERING INFORMATION

Part Number	Description
SBC 286/100	MULTIBUS II Single Board Computer

iSBC® MEM/312, 310, 320, 340 CACHE-BASED MULTIBUS® II RAM BOARDS

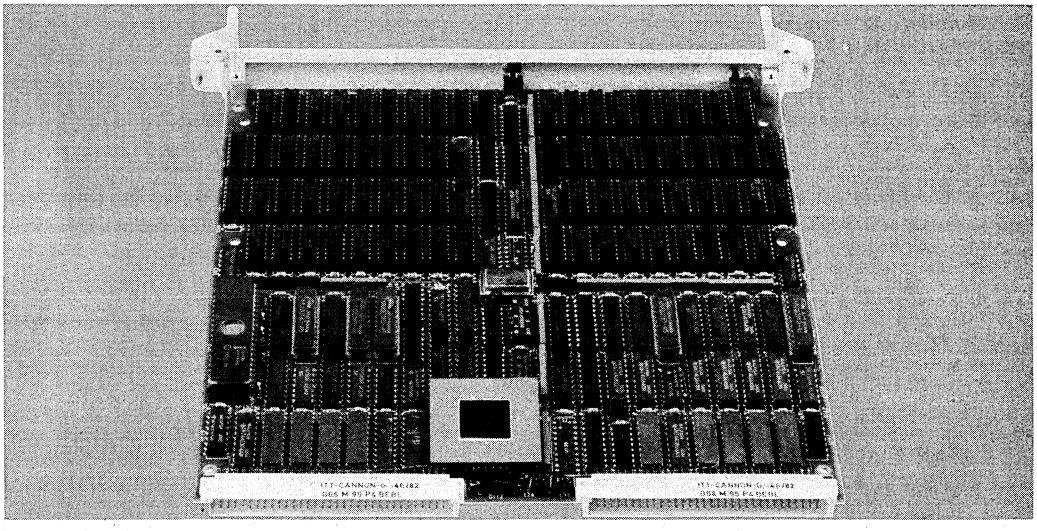
- **iSBC® MEM/3XX MULTIBUS® II memory boards are high-speed cache-based boards with 8 KBytes of cache RAM**
- **32-bit MULTIBUS® II Parallel System Bus (iPSB) and Local Bus Extension II (iLBX™ II Bus) interface support**
- **Zero wait state over iLBX™ II on a cache hit, one wait state for cache misses and writes at 8 MHz**
- **Dual port memory with four versions available:**

iSBC® MEM/312	**1/2 MBytes**
iSBC® MEM/310	**1 MByte**
iSBC® MEM/320	**2 MBytes**
iSBC® MEM/340	**4 MBytes**

- **Double-high Eurocard standard form factor, pin and socket DIN connectors**
- **MULTIBUS® II software interconnect support for dynamic memory configuration and diagnostics with no jumpers necesssary on the board**
- **Built-In-Self-Test (BIST) diagnostics on-board with both LED indicator and software access to error information**
- **Automatic memory initialization at power-up and at power-fail recovery**
- **Byte Parity error detection**

The iSBC® MEM/312, 310, 320, 340 cache-based memory boards are the first Intel memory products to implement the MULTIBUS® II system architecture. They have 32-bit architecture throughout, supporting 8-, 16-, and 32-bit central processors. The iSBC MEM/3XX (generally refers to this family of boards) memory boards are dual-ported, with access to the interfaces of both the MULTIBUS II Parallel System Bus (iPSB bus) and the iLBX™ II (Local Bus Extension).

In addition to the 32-bit memory transfer, the iSBC MEM/3XX high-speed cache control subsystem, standard on these boards, improves performance by allowing zero wait state read access over the iLBX II when data requested is in the cache memory.



SEPTEMBER, 1984
ORDER NUMBER 280071-001

FUNCTIONAL DESCRIPTION

General

The iSBC® MEM/312, 310, 320, 340 high-speed cache-based memory boards are physically and electrically compatible with the MULTIBUS® II iPSB bus standard and the new iLBX™ II bus (Local Bus Extension) as outlined in the Intel MULTIBUS II specification. Figure 1 illustrates a typical multiprocessing MULTIBUS II system configuration.

Architecture

The four main subsystems of the iSBC MEM/3XX boards are the cache controller subsystem, the cache memory subsystem, the DRAM memory subsystem, and the interconnect space subsystem (see Figure 2). The following sections describe these subsystems and their capabilities in more detail.

Cache Memory Capabilities

The cache memory system is designed around the 32-bit architecture of the main memory system. The cache memory controller and the 8 KBytes SRAM cache memory subsystems improve CPU to RAM execution to zero wait state read accesses over the iLBX II bus when data requested is in the cache memory (cache hit). A cache hit takes 125 nsec with a 8 MHz iLBX II clock.

Each entry in the 8 KByte cache memory subsystem consists of a data field of 32-bits and a tag field of up to 9-bits (depending on board DRAM size). Each byte in the main memory DRAM array directly maps to one and only one entry on the cache array. This direct mapped cache array along with tag labels ensure data integrity and accurate identification of cache hits. The cache memory size and simple but effective replacement algorithm is designed to optimize both the probability of cache hits and the CPU bus utilization. On any miss or write access, the contents of one cache entry are updated to maintain consistency with the corresponding entry in the DRAM memory array.

Dual Port Dram Capabilities

The iSBC MEM/312 module contains ½ MByte of read/write memory using 64K dynamic RAM compo-

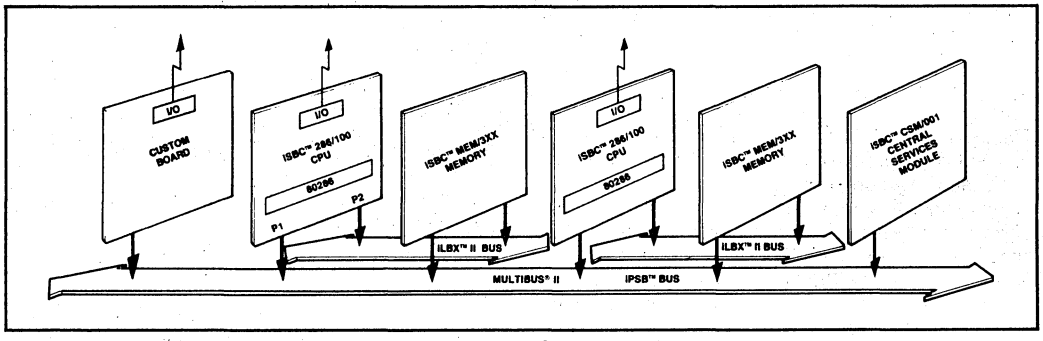


Figure 1. Typical MULTIBUS® II System Configuration

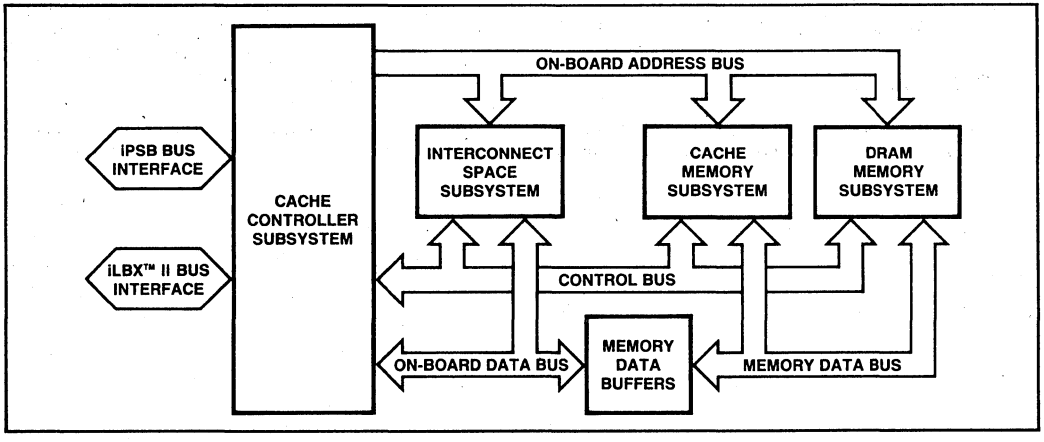


Figure 2. iSBC® MEM/3XX Board Block Diagram

nents. The iSBC MEM/310, MEM/320 and MEM/340 modules respectively contain 1 MByte, 2 MBytes and 4 MBytes of read/write memory using 256K dynamic RAM components.

The dual port capability of the iSBC MEM/3XX boards allows 32-bit access from either the iPSB bus interface or the iLBX II bus interface (see Figure 1). Due to the simple arbitration nature of the iLBX II bus interface and the cache memory subsystem, CPU to RAM memory access times over the iLBX II bus interface is optimized for up to 64 MBytes of memory.

System Memory Size

Using this series of memory boards the maximum system memory capacity based on one CPU board and 19 memory boards is 76 MBytes on the iPSB bus. (Similarly the maximum iLBX II bus memory capacity is 20 MBytes.) The memory partitioning is independent for the iPSB bus interface and the iLBX II bus interface.

The start address can be on any 64 KByte boundary on the iPSB bus and any 64 KByte boundary on the iLBX II bus. Software configures the start and ending addresses through the interconnect space. No jumpers are needed.

Interconnect Space Capabilities

The iSBC MEM/3XX board module has a set of interconnect registers which allow the system software to dynamically configure and test the status of the memory board, replacing hardwired jumper functions. This interconnect subsystem also provides control and access to the Built-In-Self-Test (BIST) features. During power-up reset, the iSBC MEM/3XX board initializes the memory and cache, sets all interconnect registers to their default values and performs a self-test. Error information from both Built-In-Self-Test (BIST) and parity checking is indicated in front panel LEDs and recorded in interconnect space registers accessible to software.

Built-In-Self-Test (BIST)

Self-test/diagnostics have been built into the heart of the MULTIBUS II system. These confidence tests and diagnostics improve reliability and reduce manufacturing and maintenance costs. LED 1 (labelled BIST) is used to indicate the status of the Built-In Self Test. It is turned on when the BIST starts running and is turned off when the BIST completes successfully. The Built-In-Self-Test performed by the on-board microcontroller at power-up or at software command are:

1. **EPROM Checksum and RAM Test:**

   This test performs EPROM checksum and RAM test on the 8751 microcontroller.

2. **Cache RAM Test:**

   This test performs RAM test on all the cache memory (HIT ONLY operation mode).

3. **Refresh Check:**

   This test performs RAM test on a small portion of DRAM with an elapsed time between the write operation and the verification of the data.

4. **Dynamic RAM Test:**

   This test performs Address Rippled RAM test on the board memory (MISS ONLY operation mode).

5. **Parity Test:**

   This test injects parity errors in the DRAM array and then verifies that the board detects these errors.

Memory Initialization and Reset

Memory is initialized automatically during power-up. All bytes are set to 00.

Error Detection Using Byte Parity

Parity will detect all single bit parity errors on a byte parity basis and many multiple bit errors. LED 2 (labelled Parity) is used to indicate parity errors. LED 2 is turned on when a parity error is detected and turned off when the parity status register within interconnect space is cleared. This same LED turns on and off during power-up to verify operation of the LED.

Error information is recorded in interconnect space so it is accessible to software for error reporting.

SPECIFICATIONS

Word Size Supported

8-, 16-, 24-, and 32-bits

Memory Size

1/2 Megabyte (iSBC MEM/312) board
1 Megabyte (iSBC MEM/310) board
2 Megabytes (iSBC MEM/320) board
4 Megabytes (iSBC MEM/340) board

Access Times (All Densities)

MULTIBUS II Parallel System Bus—iPSB (@ 10 MHz)

Read	562 ns (avg.) 775 ns (max.)
Write	662 ns (avg.) 775 ns (max.)

Note: Average access times assume 80% cache hit rates

iLBX™ II Bus—Local Bus Extension

Read	125 ns (min.) 150 ns (avg.) 250 ns (max.)
Write	250 ns (avg.) 250 ns (max.)

Base Address

iPSB Bus - any 64 KBytes boundary

iLBX II Bus—any 64 KBytes boundary

Power Requirements

Voltage—5V DC ± 5%

PRODUCT	CURRENT
iSBC MEM/312 Board	3.5 A (typ) 6.0 A (max)
iSBC MEM/310 Board	3.5 A (typ) 6.0 A (max)
iSBC MEM/320 Board	3.5 A (typ) 6.0 A (max)
iSBC MEM/340 Board	4.1 A (typ) 6.7 A (max)

Environmental Requirements

Operating Temperature: 0° to 55°C airflow of 200 linear feet per minute

Operating Humidity: To 90% without condensation

Physical Dimensions

The iSBC MEM/3XX boards meet all MULTIBUS II mechanical specifications as presented in the MULTIBUS II specification (#146077).

Double High Eurocard Form Factor:

Depth — 220mm (8.6 in.)

Height — 233mm (9.2 in.)

Front Panel Width — 20mm (.784 in.)

Weight:

iSBC MEM/312 board:	6720 gm (24 oz.)
iSBC MEM/310 board:	6160 gm (22 oz.)
iSBC MEM/320 board:	6720 gm (24 oz.)
iSBC MEM/340 board:	10080 gm (36 oz.)

Reference Manuals

iSBC MEM/3XX Board Manual (#146707-001)

Intel MULTIBUS II Bus Architecture Specification (#146077)

Manuals may be ordered from any Intel Sales Representative, Distributor Office, or from the Intel Literature Department. 3065 Bowers Ave., Santa Clara, CA 95051.

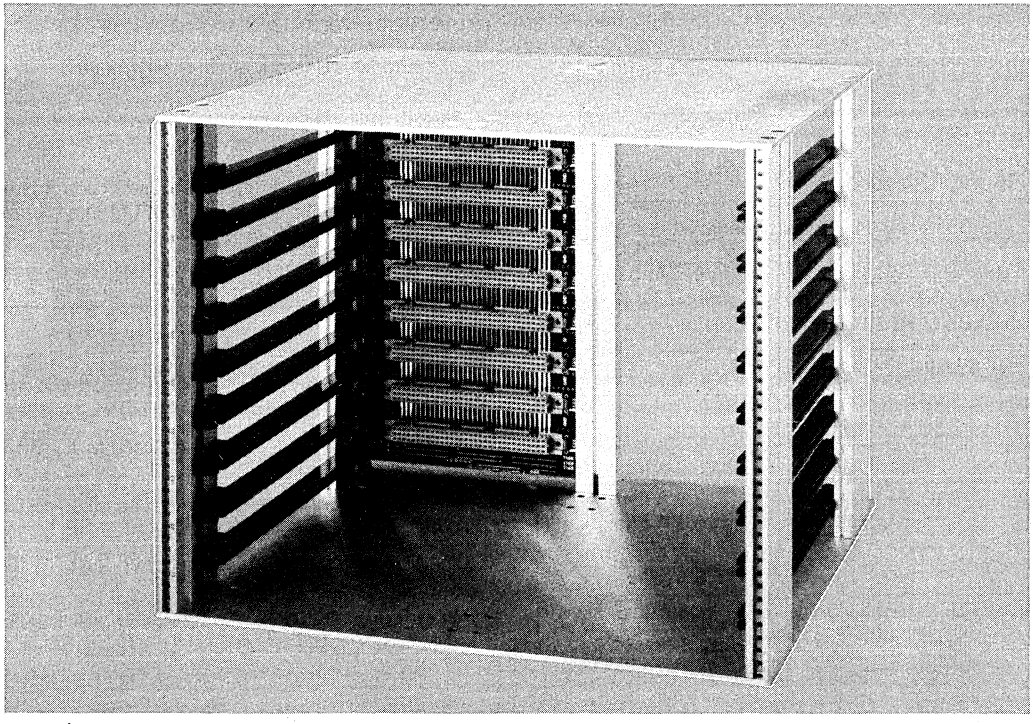
Ordering Information

Part Number	Description
iSBC MEM/312	½MByte Cache Based MULTIBUS II RAM Board
iSBC MEM/310	1MByte Cache Based MULTIBUS II RAM Board
iSBC MEM/320	2MByte Cache Based MULTIBUS II RAM Board
iSBC MEM/340	4MByte Cache Based MULTIBUS II RAM Board

iSBC® PKG/606
iSBC® PKG/609
MULTIBUS® II CARDCAGE ASSEMBLIES

- **Available in two sizes to hold up to 6 or 9 MULTIBUS® II boards**
- **Designed to mount inside a chassis or other enclosure**
- **Uses a 6 layer Parallel System Bus (iPSB) backplane**
- **All lines fully terminated per the iPSB MULTIBUS® II specification**
- **Assembly uses aluminum extrusion construction for strength and rigidity**
- **Accommodates Intel iSBC® PKG/902 and iSBC® PKG/903 2 and 3 slot iLBX™ II backplanes**

The iSBC® PKG/606/609 series of cardcages are designed to mount and interconnect up to 6 or 9 MULTIBUS® II boards for small to medium size advanced MULTIBUS II microcomputer systems. The cardcages are compact in size and easily mount in standard or custom enclosures. Extra-wide support extrusions and heavy duty endplates help make the iSBC PKG/606/609 cardcage assemblies especially suited for installation in systems located in high vibration or high shock environments. Installed in the cardcage assembly is a 6 layer iPSB backplane that utilizes separate power and ground planes and fully terminates all signal lines. This layout minimizes system noise and ensures reliable operation even in a fully loaded, multiprocessor-based system.



SEPTEMBER, 1984
ORDER NUMBER: 280075-001

FUNCTIONAL DESCRIPTION

Mechanical Features

The cardcages accommodate up to 6 (iSBC PKG/606) or 9 (iSBC PKG/609) MULTIBUS II boards spaced at 0.8 inch centers. The assemblies are designed to hold "double high" (6U) Euro form-factor boards (233.4mm high x 220mm deep) or a mixture of "single high" (3U) and "double high" boards using additional hardware (not supplied). Each installed board is held in place by two screws supplied as part of the board retainer hardware.

The cardcage frame is built using five support extrusions and two aluminum end plates as shown in figure 1. Both cardcages are 10.5″ wide and 10.1″ deep and vary in height according to model (see specifications section).

The cardcages are designed to mount inside chassis or other enclosures and may be installed so that the MULTIBUS II boards load either horizontally or vertically in the unit. All assembly hardware is countersunk allowing the cardcages to be mounted flush against any internal chassis surface.

A Parallel System Bus (iPSB) backplane is mounted to the P1 side of the assembly, and one or more iLBX™ II backplanes (not supplied) can be mounted to the P2 side.

Electrical Features

The iPSB backplane uses a 6 layer design with separate power and ground layers and a signal routing scheme which minimizes ringing, crosstalk, and capacitive loading on the bus. Mounted on the backplane are 6 or 9, 96-pin, female DIN connectors (depending on model), bus termination resistors, decoupling capacitors, and power terminals. Press-fit technology is used throughout. The PC board is UL recognized for flammability.

Single In-line Package (SIP) style resistors are used to terminate all address, clock, data, and control lines. Each termination consists of two resistors which connects the line to +VCC and ground. Different size resistors are used according to the type of driver connected to the line in an operating system.

The DIN type connectors are female, 96 pins, fully gold plated, and meet IEC standard 603-2-IEC-C096F. The connectors are mounted on 0.8″ centers to match Intel's iPSB (Parallel System Bus) MULTIBUS II backplanes and are keyed to ensure proper mating to the MULTIBUS II board. The connector can provide up to 9 amps of current at +5V to each MULTIBUS II board in addition to the current available over the iLBX II backplane.

Screw terminals on the backplane are provided for connection to +5V, ±12V power and ground. In addition, an extra +5V terminal is provided for connection to a backup battery for memory protection during power fail conditions. These terminals, each of which can handle up to 25 amps of current, provide a simple and highly reliable connection method to the system power supply.

The first slot position is designed to accept the Central Services Module (CSM) MULTIBUS II board. All other slots can accept any combination of MULTIBUS II boards.

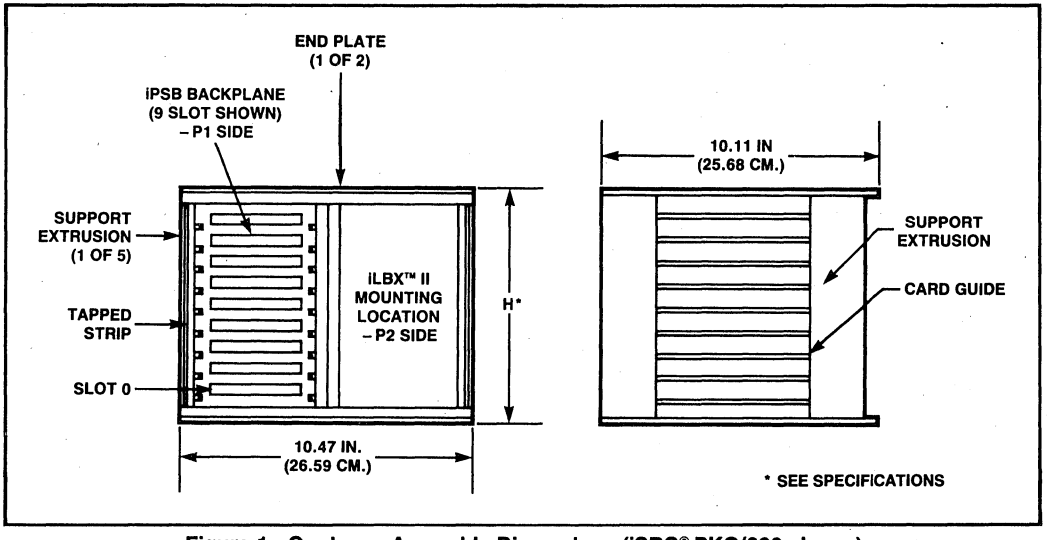


Figure 1. Cardcage Assembly Dimensions (iSBC® PKG/609 shown)

SPECIFICATIONS
Mechanical

Specification		iSBC® PKG/606 Cardcage	iSBC® PKG/609 Cardcage
Board Capacity		6	9
Dimensions	Height	5.98 in (15.20 cm)	8.38 in (21.29 cm)
	Width	10.47 in (26.59 cm)	10.47 in (26.59 cm)
	Depth	10.11 in (25.68 cm)	10.11 in (25.68 cm)
Weight		4 lbs (1.8 kg)	5 lbs (2.3 kg)
Board Spacing		0.8 in (20.3 cm)	
Mounting Hole Locations		See figure 2	
Construction Materials, Cardcage Frame		Aluminum extrusions and end plates, nylon card guides	
Construction Method iPSB Backplane		Six layer backplane with separate VCC and ground layers; all connectors, power terminals, and resistor/capacitor sockets are press-fit into the backplane	
Connector Type		96 pin "DIN" female, gold plated, meets IEC standard 603-2-IEC-C096-F	

Electrical

iPSB Backplane — Meets Intel MULTIBUS II specification No. 146077 for board dimensions, layout, signal line termination, and transmission characteristics

Power Connections — Type: Screw terminal block, AMP P/N 55181-1, Winchester P/N 121-25698-2, or equivalent

Quantity of Power Terminals and Current Rating:

Voltage	iSBC® PKG/606 Cardcage		iSBC® PKG/609 Cardcage	
	Quantity	Current (amps)	Quantity	Current (amps)
+5	3	54	4	81
+12	1	12	1	18
−12	1	12	1	18
+5BB	1	12	1	18
GND	4	78	5	117

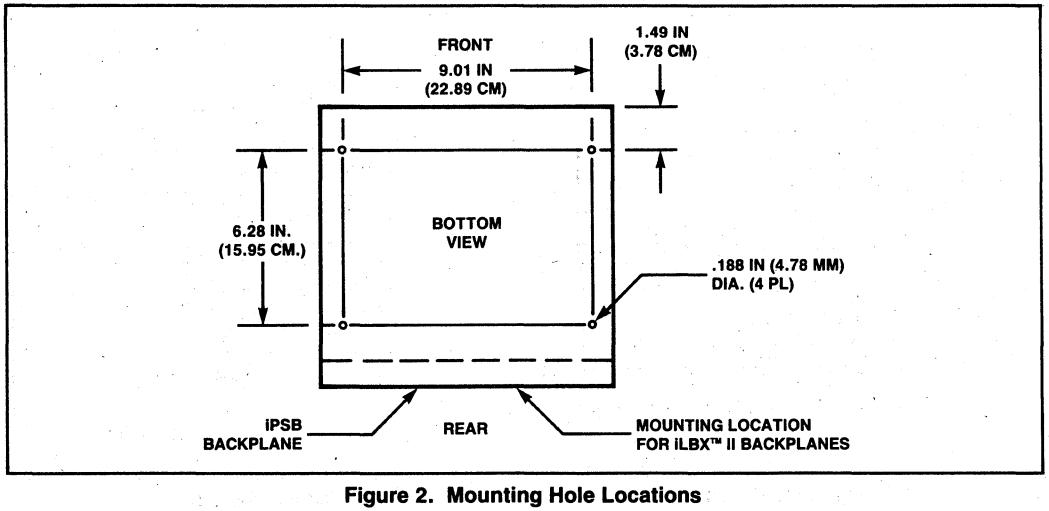


Figure 2. Mounting Hole Locations

Mating Connection: No. 6 locking spade or ring tongue lug

Maximum current available per slot:

Voltage	Current
+ 5V	9A
+12V	2A
−12V	2A
+5BB	2A

Operating Environment:
0-55°C (at 25 amps per power terminal);
0-70°C (at ≤18 amps per power terminal);
10% to 95% relative humidity, non-condensing;
0-10,000 ft. altitude

Reference Manual — MULTIBUS II Cardcage Assembly and iLBX II Backplane User's Guide, P/N 146709-001 (supplied).

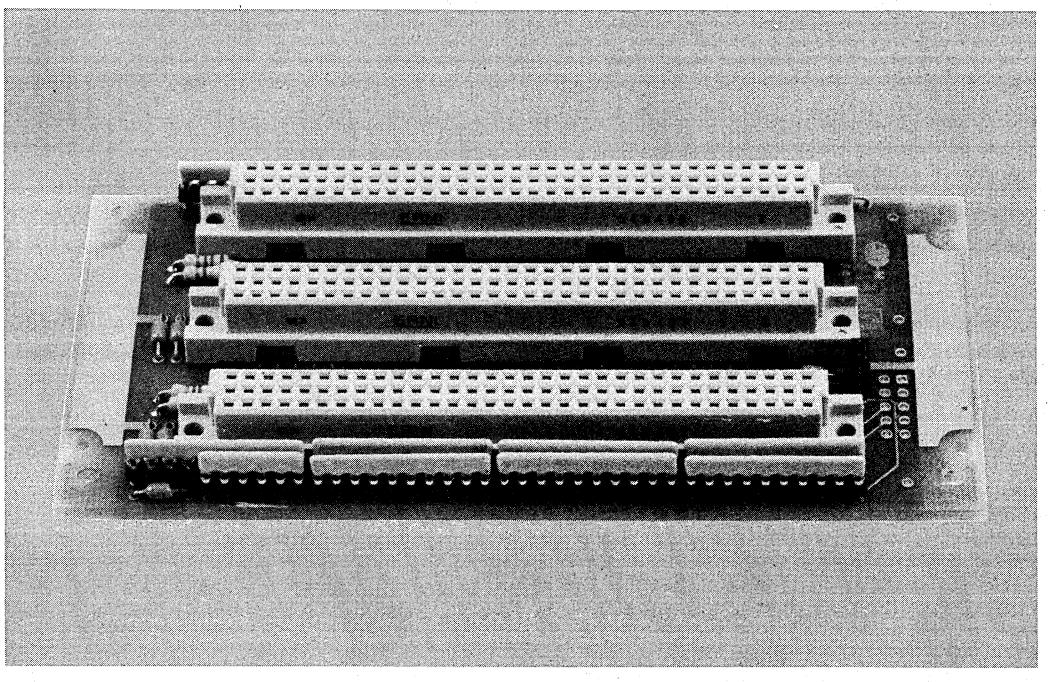
ORDERING INFORMATION

Part Number	Description
iSBC PKG/606	6 slot MULTIBUS II Cardcage Assembly
iSBC PKG/609	9 slot MULTIBUS II Cardcage Assembly

iSBC® PKG/902
iSBC® PKG/903
MULTIBUS® II iLBX™ II Backplanes

- **Provides iLBX™ II interconnect for fastest CPU/memory data transfers**
- **Designed to mount in MULTIBUS® II carcage assemblies**
- **Available in 2 slot (iSBC® PKG/902) and 3 slot (iSBC® PKG/903) sizes**
- **Uses a 6 layer, fully terminated backplane**
- **Includes a 10 pin connector for BITBUS™ applications**
- **Meets all electrical and mechanical requirements of the MULTIBUS® II specifications**

The iSBC® PKG/902 and iSBC PKG/903 series of iLBX™ II backplanes are designed to mount on the P2 side of Intel's MULTIBUS® II cardcage assembly or other double Euro (6U) cardcage. One or more backplanes may be installed in a system to allow high speed data transfers between the CPU and memory boards installed in the system. The iLBX II backplane uses a 6 layer PCB with separate power and ground planes and full termination on all signal lines. This design minimizes system noise and ensures reliable operation in all applications.



JULY, 1984
ORDER NUMBER: 280074-001

FEATURES

Mechanical and Electrical

The iSBC® PKG/902 and iSBC® PKG/903 iLBX™ II backplanes use a 6 layer printed circuit board (PCB) with separate power and ground layers and a signal lead routing scheme which minimizes ringing, crosstalk, and capacitive loading on the bus. Mounted on the PCB are two (iSBC PKG/902) or three (iSBC PKG/903) 96 pin DIN connectors, one 10-pin BITBUS™ connector, terminating resistors, decoupling capacitors, and power terminals. The resistors and capacitors are mounted into sockets, and all parts are press-fit into the backplane. The PCB is UL recognized for flammability.

Single In-line Package (SIP) style resistors are used to terminate all address, clock, data, and control lines. Each termination consists of two resistors which connects the line to +VCC and ground. Different size resistors are used according to the type of driver connected to the line in an operating system. The SIP style resistors help make the board compact in size and allows the designer to mount several backplanes directly adjacent to one another in a system without having to skip slots.

Mounted on the rear of the backplane is a 10-pin BITBUS connector. This connector serves as the serial communication interface for any iSBX 344 BITBUS controller boards installed in the system.

The DIN type connectors are female, 96 pins, fully gold plated, and meet IEC standard 603-2-IEC-C096F. The connectors are mounted on 0.8" centers to match Intel's iPSB (Parallel System Bus) MULTIBUS II backplanes and are keyed to ensure proper mating to the MULTIBUS II board. The connector can provide up to 6 amps of current at +5V to each MULTIBUS II board in addition to the current available over the Parallel System Bus backplane.

Screw terminals on the backplane are provided for connection to +5V power and ground. These terminals, each of which can handle up to 25 amps of current, provide a simple and highly reliable connection method to the power supply.

SPECIFICATIONS

Mechanical and Environmental

Connector Spacing — 0.8 in (20.3 cm)

Number of Slots — iSBC PKG/902: 2;
iSBC PKG/903: 3

Board Dimensions — See Figure 1

Connectors:

DIN — 96-pin female, gold plated, meets IEC standard 603-2-IEC-C096-F

BITBUS — 10-pin male, gold plated, T&B Ansley 609-1012M, or equivalent

Constructed Method — Six layer backplane with separate VCC and Ground layers

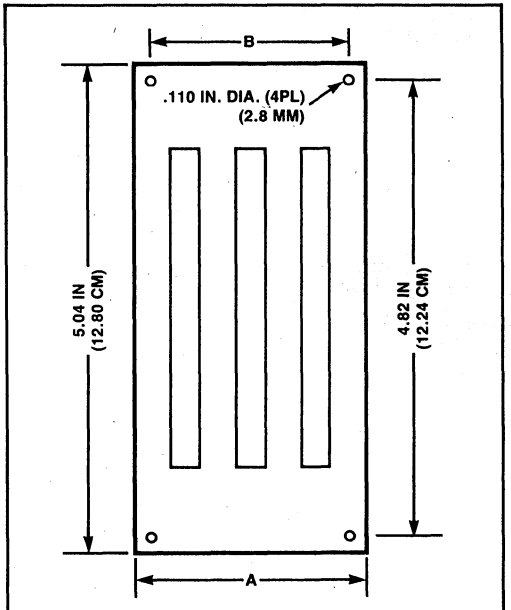
— All connectors, power terminals, and resistor/capacitor sockets are press-fit into the backplane

Mounting Hole Location — See Figure 1

Operating Environment — 0-70°C ambient temperature; 10% to 95% relative humidity, non-condensing; 0-10,000 ft. altitude

Electrical

Backplane Electrical Characteristics and Line Terminations — Per Intel MULTIBUS II specification 146077, Sec. II, iLBX II



DIMENSIONS:

		A	B
iSBC® PKG/902	IN	1.55	.80
	CM	3.93	2.03
iSBC® PKG/903	IN	2.35	1.60
	CM	5.97	4.06

Figure 1. iLBX™ II Board Dimensions (iSBC® PKG/903 Shown)

Power Connections

Type — Screw terminal block: AMP P/N 55181-1; Winchester P/N 121-25698B; or equivalent

Mating Connection — No. 6 locking spade or ring tongue lug

Quantity — 2 (VCC, Ground)

Current Rating — iSBC PKG/902: 12 amps; iSBC PKG/903: 18 amps

Maximum Current Available Per Slot — 6 amps (over the iLBX II backplane)

REFERENCE MANUAL

MULTIBUS® II Cardcage Assembly and iLBX™ Backplane User's Guide, P/N 146709-001 (not supplied).

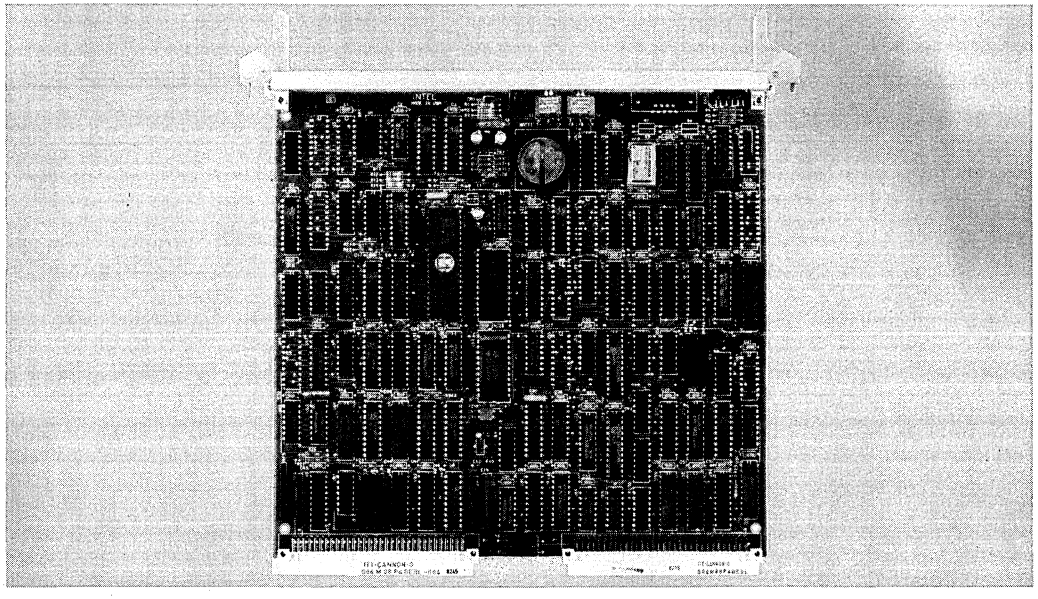
ORDERING INFORMATION

Part Number	Description
iSBC PKG/902	2 slot iLBX II Backplane
iSBC PKG/903	3 slot iLBX II Backplane

iSBC® CSM/001
CENTRAL SERVICES MODULE

- **iSBC® CSM/001 Central Services Module integrates MULTIBUS® II central system functions on a single board**
- **MULTIBUS® II Parallel System Bus clock generation for all agents interfaced to the MULTIBUS® II iPSB bus**
- **System-wide reset signals for power-up, warm start, and power failure/recovery**
- **System-wide time-out detection and error generation**
- **Slot I.D. and Arbitration I.D. initialization**
- **MULTIBUS® II interconnect space for software configurability and diagnostics**
- **Built-In Self Test (BIST) power-up diagnostics with LED indicator and error reporting accessible to software via interconnect space**
- **Interface to the MULTIBUS® I Link board with necessary buffering and protocol conversion**
- **Time-of-day clock support with battery back-up on board**
- **Double-high Eurocard standard form factor, pin and socket DIN connectors**

The iSBC® CSM/001 Central Services Module is responsible for managing the central system functions of clock generation, power-down and reset, time-out, and assignment of I.D.s defined by the MULTIBUS® II specification. The integration of these central functions in a single module improves overall board area utilization in a multi-board system since these functions do not need to be duplicated on every board. The iSBC CSM/001 module additionally provides a time-of-day clock and the interface to the MULTIBUS I Link board.



SEPTEMBER, 1984
ORDER NUMBER 280070-001

FUNCTIONAL DESCRIPTION

Overall

The iSBC® CSM/001 Central Services Module integrates MULTIBUS® II central system functions on a single board. Each MULTIBUS II system requires management of these central system functions as defined in the MULTIBUS II specification. Figure 1 illustrates a typical multiprocessing MULTIBUS II system configuration. To perform its central system functions, the iSBC CSM/001 Central Services Module has a fixed slot I.D. and location in the backplane. The iSBC CSM/001 board additionally provides an interface to the MULTIBUS I Link board and a time-of-day clock.

Architecture

The iSBC CSM/001 board is functionally partitioned into 6 major subsystems. The Centralized System Wide Control subsystem includes MULTIBUS II iPSB bus clock generation and system wide reset signal generation. The Time-Out Control subsystem provides system wide time out detection and error generation. The System Interconnect Space subsystem controls I.D. initialization and software configurable interconnect space. The Link Board interface subsystem provides an interface to the MULTIBUS I Link board. The last two subsystems are the Time-of-Day clock and the iPSB bus interface. These areas are illustrated in Figure 2.

CENTRALIZED SYSTEM-WIDE CONTROL SUBSYSTEM

Parallel System Bus Clock Generation

The CSM generates the Parallel System Bus clocks. The Bus Clock (BCLK*) 10MHz signal and the Constant Clock (CCLK*) 20MHz signal are supplied by the CSM to all boards interfaced to the Parallel System Bus. These boards use the Bus Clock 10 MHz signal for synchronization, system timing, and arbitration functions. The Constant Clock is an auxiliary clock. The frequency of the Bus Clock and Constant Clock can be halved via jumpers for diagnostic purposes.

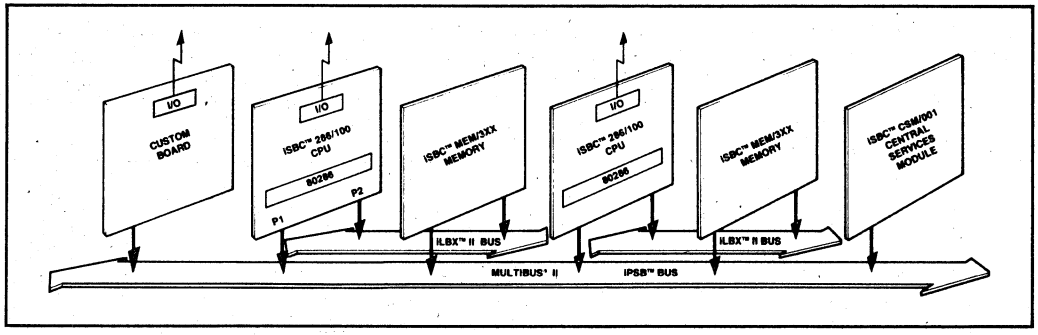


Figure 1. Typical MULTIBUS® II System Configuration

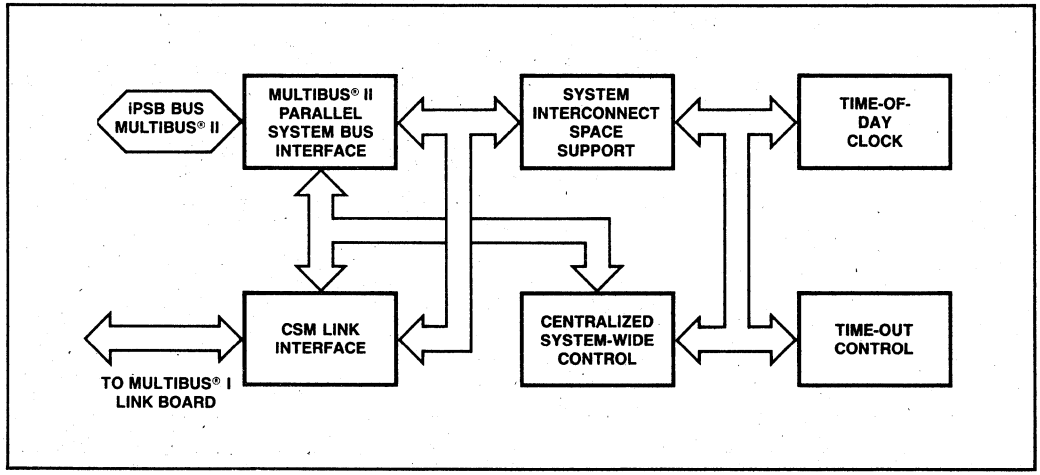


Figure 2. Block Diagram of iSBC® CSM/001 Board

Reset Control and Power-Fail/Recovery

The CSM sends a system-level reset/initialization signal to all boards interfaced to the Parallel System Bus. The CSM assigns slot I.D. and arbitration I.D. to these boards during this initialization process. It provides this signal upon pressing of the reset switch, restoration of system power or a software request for reset received via the CSM interconnect space. The reset switch may be jumper-configured to cause a power-up or warm reset, with warm reset the default configuration. The reset switch is located on the front panel. Additionally, warm reset and cold reset signals can be sent through the P2 connector.

The CSM power supply interface is accomplished via the ACLO input of the P2 connector. ACLO is an open collector input from the power supply which provides advance warning of imminent power fail. If battery backup is not required, a jumper is provided on the CSM to disable the power fail signal ACLO.

TIME-OUT SUBSYSTEM

The TIMOUT* (Time-out) signal is provided by the CSM whenever it detects the failure of a module to complete a handshake. This TIMOUT* signal is received by all boards interfaced to the iPSB bus and may be disabled via the interconnect space.

INTERCONNECT SUBSYSTEM

The CSM Interconnect subsystem provides arbitration I.D., and slot I.D. initialization, software configurable interconnect space, and on-board diagnostics capability.

At reset, the CSM supplies each board interfaced to the iPSB bus with its slot I.D. and its arbitration I.D. The slot I.D. assignment allows user or system software to address any board by its physical position in the backplane.

The interconnect space has both read-only and software configurable facilities. The read-only registers hold information such as vendor number, board type, board revision and serial number so that this information is available to the system software. The CSM software configurable interconnect space allows write operations to support board configuration and diagnostics under software control. The CSM also uses interconnect space for system wide functions such as providing a time/date record (from time-of-day clock), software access to diagnostics and software control of the system wide functions.

BUILT-IN-SELF-TEST (BIST) DIAGNOSTICS

Self-test/diagnostics have been built into the heart of the MULTIBUS II system. These confidence tests and diagnostics improve reliability and reduce manufacturing and maintenance costs. LED 1 (labeled BIST) is used to indicate the status of the Built-In-Self-Test. It is turned on when the BIST starts running and is turned off when the BIST completes successfully. In addition, all error information is recorded in interconnect space so it is accessible to software for error reporting.

The Built-In-Self-Tests performed by the on-board microcontroller at power-up or at software command are:

1. PROM Checksum Test—Verifies the contents of the 8751 microcontroller.
2. RAM Test—Verifies that each RAM location of the 8751 microcontroller may store 0's and 1's by complementing and verifying twice each RAM location.
3. Real Time Clock Chip RAM Test—Verifies that reads and writes to the RAM locations on Real Time Clock Chip are functional
4. Real Time Clock Test—Reads and writes all RAM locations of the RTC chip. Not run at power-up due to destructive nature.
5. Arbitration/Slot I.D. Register Test—Verifies that arbitration and slot I.D.s can be read and written from on-board.
6. 8751 Status Test—Verifies that input pins of the 8751 are at correct level.
7. Clock Frequency Test—Tests accuracy of Real Time Clock to .2% against bus clock.

CSM LINK INTERFACE

The CSM Link Interface and the MULTIBUS I iSBC Link board provides a bridge between MULTIBUS I and MULTIBUS II systems. Hybrid systems can be built for development or target. The CSM Link Interface uses the P2 connector on the iSBC CSM/001 module for transferring commands and data from MULTIBUS II to a MULTIBUS I Link board. The MULTIBUS I Link board is purchased separately from the iSBC CSM/001 board and includes the cable which connects the iSBC CSM/001 board and the MULTIBUS I Link board (see Figure 3).

The CSM Link Interface supports 8 or 16-bit transfers via a 16-bit address/data path. The iSBC Link board resides in the MULTIBUS I system and provides a memory and I/O access window to MULTIBUS I from the MULTIBUS II Parallel System Bus. Only one iSBC Link board can be connected to the iSBC CSM/001 module.

TIME-OF-DAY CLOCK SUBSYSTEM

The Time-Of-Day Clock subsystem consists of a clock chip, battery, and interface circuitry. The clock provides time keeping to 0.01% accuracy of fractions of seconds,

seconds, minutes, hours, day, day of week, month, and year. This information is accessible via the interconnect space. The battery back-up for the clock chip provides 2 years of operation.

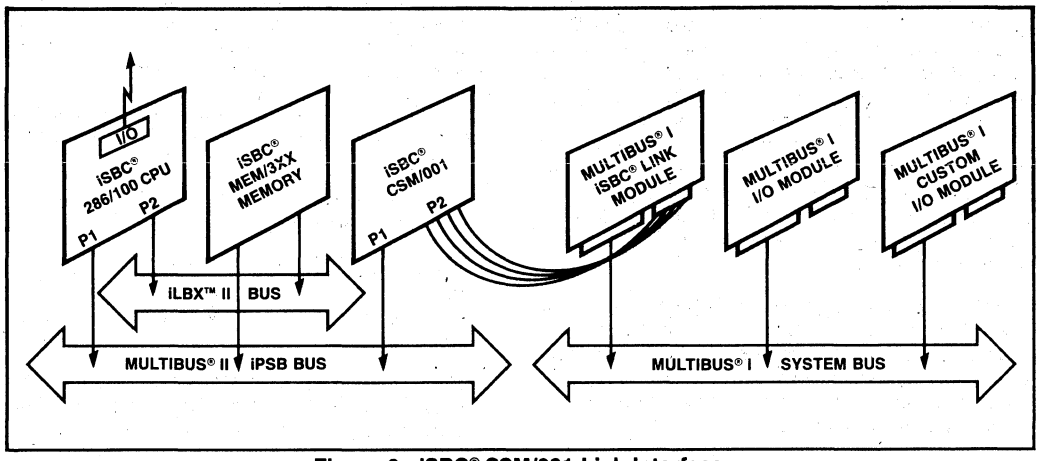


Figure 3. iSBC® CSM/001 Link Interface

SPECIFICATIONS

System Clocks

BCLK* (Bus Clock)	10 MHz
CCLK* (Constant Clock)	20 MHz
LCLK* (Link Clock)	10 MHz

Jumper option available to divide these frequencies in half

Interface Compliance

MULTIBUS II Bus Architecture Specification (#146077)

Link Cable

The Link cable uses a 64 conductor ribbon cable for interconnecting the CSM board to the Link Board. The maximum length for the cable is 1 meter.

Interface Specifications

Location	Function	Part #
P1	iPSB Bus	603-2-IEC-C096-F
P2	Link and Remote Services	603-2-IEC-C064-F

PHYSICAL DIMENSIONS

The iSBC CSM/001 board meets all MULTIBUS II mechanical specifications as presented in the MULTIBUS II specification (#146077).

Double-High Eurocard Form Factor:

Depth—220mm (8.6 in.)

Height—233mm (9.2 in.)

Front Panel Width—20mm (.78 in.)

Weight—4820 gm (16.5 oz.)

ENVIRONMENTAL REQUIREMENTS

Operating Temperature: 0° to 55°C airflow at 200 linear feet per minute

Operating Humidity: 50 to 95% without condensation (25° to 55°C)

POWER REQUIREMENTS

Voltage (volts)	Current (amps)
+5V	3A (max.)
+5VBB	1A (max.)

BATTERY CHARACTERISTICS

3V nominal voltage; capacity of 160 milliamp hours minimum.

BATTERY DIMENSIONS

Outside dimension	20mm-23mm
Height	1.6mm-3.2mm

REFERENCE MANUALS

iSBC CSM/001 Board Manual (#146706-001)

Intel MULTIBUS II Bus Architecture Specification (#146077)

Manuals may be ordered from any Sales Representative, Distributor Office, or from the Intel Literature Department, 3065 Bowers Ave., Santa Clara, CA 95051

ORDERING INFORMATION

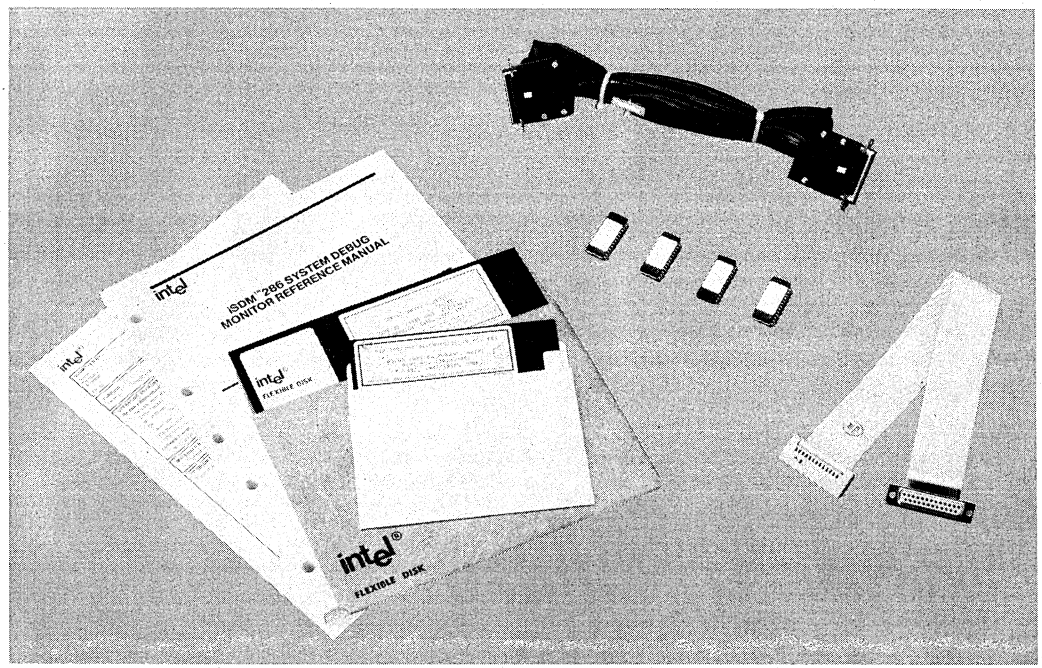
Part Number	Description
iSBC CSM/001	MULTIBUS II Central Services Module

PRELIMINARY

iSDM™ 286
iAPX 286 SYSTEM DEBUG MONITOR

- **Development support for iSBC® 286- and iAPX 286-based applications**
- **Real Address Mode (RAM) and Protected Virtual Address Mode (PVAM) support**
- **Support of MULTIBUS® I and MULTIBUS® II environments**
- **Powerful debugging commands, including single step CPU operation**
- **For MULTIBUS® II, software configuration of system boards at start-up and automatic configuration of memory boards**
- **Universal Development Interface (UDI) support via development system connection**
- **Command execution, including program load capability from Intellec® Series III or Series IV Development Systems**
- **Supports 80287 Numeric Processor Extension (NPX) for high-speed math applications**

The Intel iSDM™ 286 System Debug Monitor package contains the necessary software, cables, EPROMs, and documentation required to interface an iSBC® 286 board or iAPX 286 application to an Intellec® Series III or Series IV through a high-speed link. The System Debug Monitor supports an OEM's choice of MULTIBUS® I or MULTIBUS II environments, and the iRMX™ 86 Real-Time Multitasking Operating System or a custom operating system. The monitor contains debugging tools that examine CPU registers, memory content, CPU descriptor tables, and other crucial environmental details. The Monitor also allows programs to access files on the development system via the internal UDI support and the serial communication link.



SEPTEMBER, 1984
ORDER NUMBER: 230882-002

FUNCTIONAL DESCRIPTION

Overview

The iSDM 286 System Debug Monitor provides programmers of iAPX 286-based applications with the debugging tools needed to test new applications ranging from single-user systems to complex operating systems executing in either a MULTIBUS I or MULTIBUS II environment. Programmers are given direct access to both the Real Address (RAM) and Protected Virtual Address (PVAM) modes of the CPU via a simple terminal interface or via an Intellec Series III or Series IV Development System.

Powerful Debugging Commands

The iSDM 286 Monitor contains a powerful set of user functions, including commands to:

- Examine and modify CPU registers
- Examine, modify, and move memory locations
- Symbolic reference to variable names
- Find and compare memory contents
- Set program breakpoints
- Bootstrap load application software from iRMX 86 file compatible peripherals (requires the iRMX 86 Operating System for Bootstrap Loader)
- Single-step CPU operation
- Switch from Real Address Mode to Protected Virtual Address Mode

Formatted Displays

The iSDM 286 Monitor formats all iAPX 286 predefined data structures into clearly understandable displays. This display gives programmers a formatted view of such CPU structures as LDTs, GDTs, IDTs, Segment Selectors, and Task State Segments—not just a series of unconnected digits.

Universal Development Interface (UDI)

Via the Universal Development Interface (UDI), the iSDM 286 Monitor can support the execution of iRMX 86, Series III, Series IV, or any other UDI-based applications. The Monitor emulates many of the UDI calls (RAM or PVAM), and passes all requests for a file system to the host development system. UDI applications, such as compilers and other programs available from Independent Software Vendors, can be tested in the target iAPX 286 environment immediately.

MULTIBUS® II Software Configuration of System Boards

The MULTIBUS II Interconnect Space Registers allow the software to configure boards, eliminating much of the need for jumpers and wire wraps. The iSDM 286 Monitor can initialize these registers at configuration time using user-defined variables. The Monitor can also automatically configure memory boards, defining the addresses for each board sequentially in relation to the board's physical placement in the card cage. This feature allows for the swapping, adding, and deleting of memory boards on a dynamic basis.

Command Execution

Commands to the iSDM 286 Monitor are entered interactively via a standalone terminal, an Intellec® Series III or a Series IV Development System. The target application hardware is connected to the terminal or development system via a serial link. Figure 1 shows a typical MULTIBUS I environment and Figure 2 shows a typical MULTIBUS II environment. All control operations and UDI file manipulations occur over the serial link through the cables supplied. More than one channel can be configured for the communication since the Monitor scans all configured channels to determine which channel is in use.

Numeric Data Processor Support

In addition to executing 80287 Numeric Processor Extension (NPX) applications with full NPX performance, programmers may examine and modify NPX registers using decimal and real number format. Any location in memory known to contain numeric values in standard real format (IEEE-P754) may be examined or modified using normal decimal notation. In this manner, programmers may feel confident that correct and meaningful numbers are available to applications without having to encode and decode complex real, integer, and BCD hexadecimal formats.

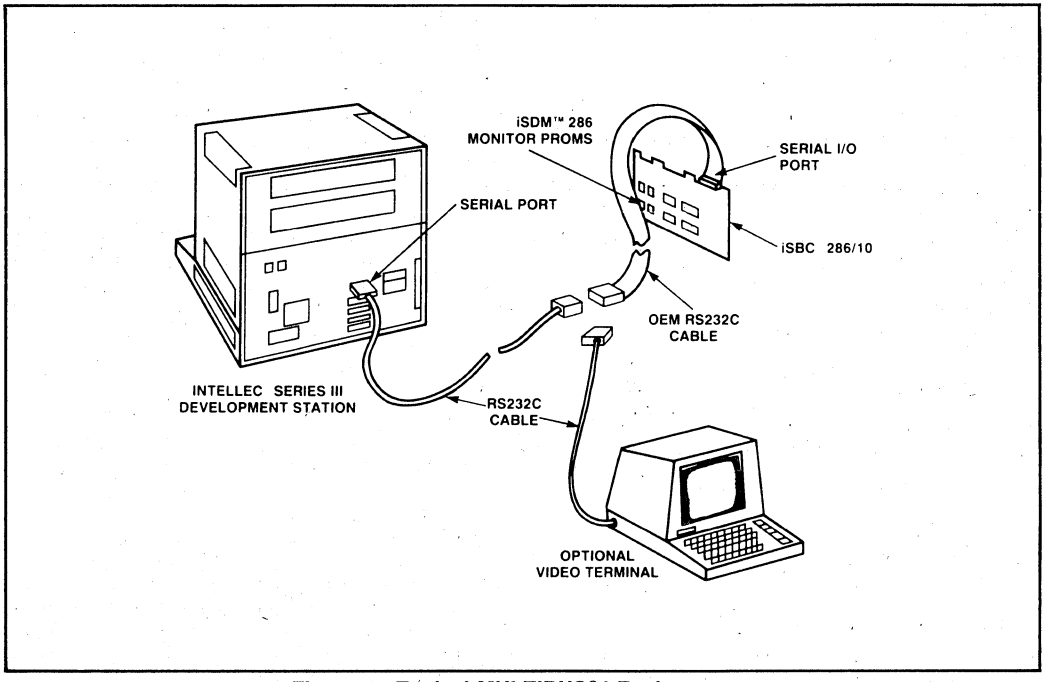


Figure 1. Typical MULTIBUS® I Environment

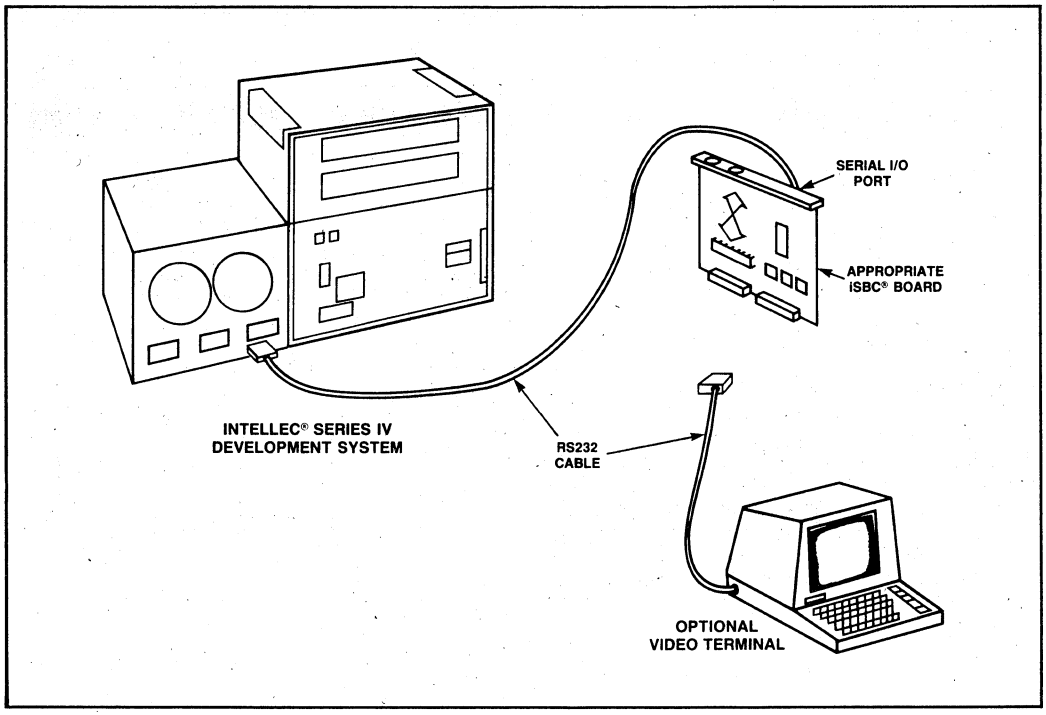


Figure 2. Typical MULTIBUS® II Environment

SPECIFICATIONS

Development System Environment

Intellec Series III or Series IV Development System with 128K of memory and 1 disk drive.

Target System Environment

Any iAPX 286 system with at least 4K of read-write memory starting at location 0H and 32K of read-only memory starting at location 0FF8000H.

Serial communication with a stand-alone terminal or development system requires either a 8274 USART and 8253 or 8254 PIT, or an 82530 SCC.

Monitor EPROMs are supplied for locations 0FF8000H through 0FFFFFFH.

ORDERING INFORMATION

The iSDM 286 System Debug Monitor package includes cables, EPROMs, software, and a reference manual. The software is provided on a double-density, single-sided ISIS-formatted 8″ diskette for Series III Development System use and on a double-density, double-sided iRMX-formatted 5¼″ diskette for Series IV Development System use.

The OEM license option listed here allows users to incorporate iSDM 286 into their applications. Each use requires payment of an Incorporation Fee.

ORDER CODE: iSDM 286 RO.

The iSDM 286 RO product also includes 90 days of support services that includes the Software Problem Report service.

Another licensing option includes prepayment of all future incorporation fees.

As with all Intel software, purchase of any of these options requires the execution of a standard Intel Master Software license. The specific rights granted to users depends on the specific option and the license signed.

intel®

ARTICLE REPRINT

AR-356

October 1984

Microprocessor Bus is Ready to Meet 32-Bit Applications of Future

JOHN BEASTON

Order Number: 280081-001

Multiprocessor bus is ready to meet 32-bit applications of future

Using five buses, Multibus II promises to handle the high speed and large I/O applications being developed for divergent microprocessors

by John Beaston, *Intel Corp., Hillsboro, Ore.*

☐ Board vendors who want to take advantage of the latest in microprocessor technology no longer have incompatible bus lines and interfaces blocking their paths. Multibus II, which was designed to support systems containing practically any microprocessor, has cleared the way. Moreover, because of its functional partitioned architecture, the latest chips or boards can be easily substituted for their older counterparts without system disruption. In addition, easily expandable inputs and outputs can meet the needs of most applications.

Multibus II's bus architecture takes the multiple-bus structure for microprocessors of Multibus I and expands its available system bandwidth, multiprocessing support, reliability, and cost-performance range, while making the architecture easy to use. Each of its five buses (see "Bus quintet brings networking flexibility," opposite) includes facilities or advanced bus structures that anticipate the needs of processors into the 1990s.

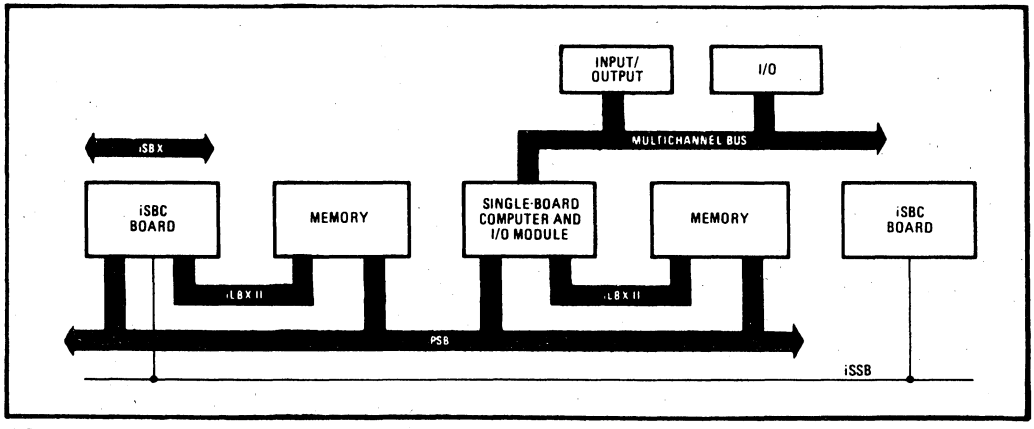
A bus structure consists of the bus's electrical and mechanical characteristics. It includes such things as specifications for the connector pin-out, the control-line protocol, the voltage levels, and current drive. Each bus has slightly different structural characteristics.

The bus architecture, on the other hand, refers to the actual systems that can be built with the bus or buses. A single, simple bus usually means low or moderate performance and limited peripheral expansion. Multiple buses, however, give system designers greater flexibility in meeting the system requirements.

The Multibus II (Fig. 1) has five buses: the parallel system bus (iPSB); the local bus extension (iLBX II); the serial system bus (iSSB); and two buses borrowed from the Multibus I architecture, the iSBX input/output expansion bus and the Multichannel direct-memory-access bus. These buses can be combined in almost any way.

The iPSB is the system's main communication path and provides the complete set of system functions: read/write accesses to memory and I/O, intermodule communication, arbitration between nodes contending for the bus, and central functions such as system reset and power-down. It can support 8-, 16-, and 32-bit processors and both 32-bit address and 8-, 16-, 24-, and 32-bit data transfers. Using 32-bit data-and-burst transfers, a 40-megabyte-a-second transfer rate can be achieved using existing technology for the bus interface.

The iPSB bus is highly multiplexed. The 32-bit address



1. Five buses. Multibus II uses five buses: the iSBX input/output expansion bus, the Multichannel DMA bus, the parallel system bus (iPSB), the local bus extension (iLBX II), and the serial system bus (iSSB). With this architecture, boards or chips can be swapped easily.

Bus quintet brings networking flexibility

At the heart of the Multibus II specification lie its five buses, three of which are new. The parallel system bus, the local bus extension, and the serial system bus are the newly created buses; the iSBX input/output expansion bus and the Multichannel direct-memory-access bus are borrowed from Multibus I.

The parallel system bus or the iPSB is a high-performance, general-purpose bus that provides important data-movement and interprocessor-communication functions. It supports arbitration, execution, and I/O data-movement and gives board-configuration support. This bus has four address spaces: a 32-bit-wide memory address space, a 16-bit I/O address space, a 16- or 32-bit message address space, and a 16-bit interconnect address space. Data is clocked at 10 megahertz and can be up to 32 bits wide. In addition, the iPSB's burst-transfer capability yields a maximum sustained bandwidth of 40 megabytes a second.

Multiple processors executing an instruction in shared global memory can easily saturate the system bus and degrade system performance overall. The iLBX II, or local bus extension, can remove the processor's execution functions from the general-purpose system bus and extend local on-board performance to off-board memory resources. The bus provides up to 64 megabytes of arbitration-free local-memory expansion and a 12-MHz clock rate. The iLBX II bus also has a large bandwidth of 48 megabytes/s and advanced features such as pipelining and block transfers.

The iSSB, or serial system bus, is a low-cost serial interconnection that functions as an alternative to the message interface on the iPSB. Whereas the iPSB has 32-bit-wide parallel transfers and runs at 10 MHz, the iSSB is 1 bit wide and runs at 2 MHz. In addition, while the iPSB has 96 pins, the iSSB has only 2. Finally, because of the iSSB's serial structure, it can be extended up to 10 meters, whereas its parallel counterpart must be rigidly packaged.

Carried over from Multibus I, the iSBX I/O bus moves high-speed I/O data to and from physically distributed custom peripherals such as mass-storage devices or graphics-display systems. Block transfers at 8 megabytes/s between peripherals and single-board computers are possible. In addition, the bus provides a standardized I/O interface with full-speed operation at up to 15 meters with a simple asynchronous protocol. The bus supports up to 16 processors, both 8- and 16-bit, and allocates 16 megabytes of memory or register address space to each.

Finally, the iSBX I/O expansion bus is used for incremental on-board system expansion through small iSBX multimodule boards. Currently, iSBX boards add capabilities like parallel I/O, serial I/O, graphics, and advanced mathematical functions. All iSBX boards afford system expansion without the requirement of adding another expansion board. Through this bus, users can customize their single-board computers to individual applications in response to the latest very large-scale-integration technology. Since users are able to buy exactly the capabilities needed, both system cost and size are minimized.

---

and data paths are multiplexed, and the nine system-control lines have different uses depending on the transfer cycle's phase. The six arbitration lines serve the dual purposes of system initialization and normal operation.

Compared with nonmultiplexed buses, this multiplexed structure significantly reduces the number of lines in the bus interface. The benefits of doing this are many: pin efficiency, interface power requirements, and ease of trace routing. The iPSB bus is so efficient with its pin placement that the entire 32-bit iPSB bus fits into a single 96-pin connector. Even the smaller, single-connector form-factor boards contain a full 32-bit interface.

Multiplexing reduces a board's total current requirements and thus the amount of noise generated. (Nonmultiplexed 32-bit buses require at least 64 high-current drivers.) The multiplexed structure's fewer lines make trace routing easier, which, in turn, leaves more board area available for added functionality.

The simplest system may be built with the iPSB bus only. If the system is execution-intensive, an iLBX II bus can extend the processor's local bus to off-board memory resources and maintain the same performance as if the memory were local. In fact, since the iLBX II bus is connected to a given processor, the system might contain as many iLBX II buses as there are processors.

The variety of bus structures in the Multibus II architecture has five significant benefits: increased system bandwidth, improved reliability, reduced service costs and easy configuration, enhanced multiprocessing support, and a better price-performance ratio.

Big bandwidths

The 96-megabyte/s system bandwidth available with Multibus II is the highest among today's standard bus architectures and is due to the concurrent operation of the buses (Table 1). Because the buses are independent, one processing module can execute code over its iLBX II bus while another transfers data over the iPSB bus. The same is true for the iSSB and Multichannel buses.

Thus a system having a single iPSB bus, an iLBX II bus, and a Multichannel bus has a bandwidth of 96 megabytes/s. The iSBX-bus bandwidth is not counted in the total because it is a specialized on-board I/O extension.

Because of its large bandwidth, the system can accommodate many processing modules without the buses becoming saturated. Because intelligence is being used increasingly in functional modules, this extra bandwidth gives the designer the space needed for each module added to improve system performance.

The unusually large bandwidths of the iPSB and iLBX II

TABLE 1 MULTIBUS II BANDWIDTHS

Bus type	Bandwidth (megabytes/sec)
iPSB	40
iLBX II	48
iSBX	10
iSSB	2 (M/sec)
Multichannel	8

TABLE 2 iSSB BUS SIGNAL ENCODING

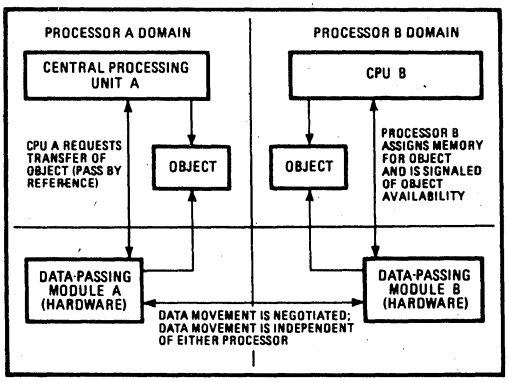
Line A	Line B	State
0	0	collision
0	1	logic 0
1	0	logic 1
1	1	idle

buses—40 and 48 megabytes/s, respectively—is due to their ability to support sequential or block-mode transfers. The 32-bit data paths are important since advanced 32-bit processors should be readily available soon, and the increased data width doubles the bandwidth over comparable 16-bit buses. Sequential transfers improve bus utilization because only one address is used for multiple data cycles (the replying modules automatically increment the starting address).

The multiple-bus structure increases system reliability by increasing connector reliability, data-transfer integrity, and electrical reliability. Multibus II boards use the popular two-piece DIN (IEC 603-2) connector. This connector is part of the Eurocard mechanical standard. The 96-pin version used by Multibus II is noted for its exceptional reliability and is available from numerous sources.

Another aspect of system reliability concerns detection of errors occurring during data transfers, and each of the new Multibus II buses contain mechanisms to detect such errors. The parallel iPSB and iLBX II buses are parity-protected, while the serial iSSB bus uses cyclic redundancy checks. As a result, every data transfer on these buses is completely protected.

In addition, the buses protect the addresses and commands. The detecting modules signal an error's occurrance on the bus, and the appropriate bus-interface logic notifies the involved modules to correct the problem.



2. Message passing. The message-passing scheme for Multibus II is its strongest attribute and adds significant multiprocessing support. The two communicating message-passing interfaces acts as a distributed intelligent DMA controller.

To improve electrical reliability, the parallel buses use synchronous clocking in order to increase immunity from crosstalk and noise. All control, address, and data lines are sampled on a clock edge. Because only one edge of the clock line is important, edges appearing on other lines from noise or crosstalk are ignored, which greatly improves electrical reliability over the more traditional buses that use edge-sensitive asynchronous protocols. Furthermore, synchronous clocking simplifies system design and debugging: the bus interface is easily controlled through state machines, and logic analyzers are easily adapted for debugging.

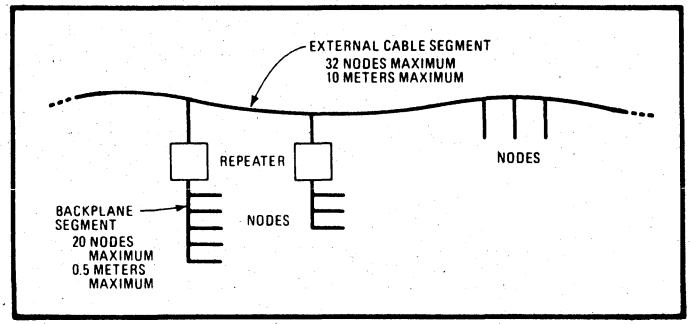
Configuration ease

Board suppliers tend to produce products that address many diverse needs so the boards are usually loaded with jumper options and switches in dual in-line packages. But cramming all these options on a board can lead to confusion and configuration errors.

However, the iPSB and iLBX II define a mechanism, called geographical addressing, that can eliminate or reduce many of these manual programming options while maintaining flexibility. With it, software can address a board through its physical position in the backplane. Interconnect registers, placed on each board, contain such information as the manufacturer, the board type, and revision level. Software can thus read these registers and identify which boards are in the system and where. It can configure the board by writing into these registers. Changing a memory board's starting address, for example, is as simple as writing to that board's appropriate starting-address register.

The interconnect register opens the door to improved diagnostics performed either locally or remotely. A designer may build in a modem with which service technicians can exercise the system at the customer's site. Technicians can remotely determine the system's exact makeup and either download or trigger the appropriate diagnostic software locally.

The Multibus II's support for multiple processors goes beyond increased system bandwidth. The iPSB bus-arbitration and -interrupt structures are tailored for multiprocessing and, most importantly, the multiprocessing specification includes a message-passing protocol.



3. Serial setup. The iSSB serial system bus can have up to 32 external nodes distributed over a 10-meter maximum, or 20 nodes clustered into backplanes connected to the bus cable by repeaters that isolate the bus from the excessive backplane capacitive load.

A bus's arbitration structure determines how many master modules can reside simultaneously on the bus. The iPBS allows up to 20 bus masters, whereas standard buses support a handful. In fact, every board in a Multibus II system can be a bus master. Two arbitration algorithms are available: fairness and high priority. The fairness algorithm gives every board a fair chance to access the bus. As a result, no board is forever locked out by a higher-priority master.

On the other hand, the high-priority algorithm can be used to define a priority structure among all the boards in the system. In this case, boards handling real-time events are usually given the highest priority, while less critical boards, such as control panels, get a lower priority. Changing between the two algorithms can be done dynamically; the designer is free to choose one or the other, or both.

Interrupts are usually a source of frustration for the system designer. Large systems contain many interrupt sources, yet most buses provide eight lines or so to handle them. As a result, several sources share a given interrupt line, and either software polling or hardware daisy-chaining is used to decide which source on that line actually generated the interrupt. More interrupt lines are needed, but most buses do not have the extra lines to spare. Using available lines as virtual interrupt lines, the iPSB's interrupt structure supports up to 255 interrupt sources and as many destinations.

Message passing

Each module on the bus is assigned a module message address. Special interrupt bus cycles let any module send an interrupt to any other module by specifying the appropriate source and destination message addresses. Another way to think of this is as memory-mapped interrupts where the source module writes into the destination's message address space to trigger an interrupt. This creates so many virtual lines that the designer does not have to worry about running out of them or having to share lines among modules.

The Multibus II message-passing protocol is probably its most significant contribution to multiprocessing. Most multiple-processor systems use either a pass-by-reference or a pass-by-value intermodule-communication protocol. In the pass-by-reference mode, the communicating modules exchange pointers or tokens so that they may address shared data structures. This method usually performs well, but it is difficult to extend beyond two processors sharing one data structure. It is also not appropriate for memory-protected operating systems, which are more likely to use the pass-by-value-method.

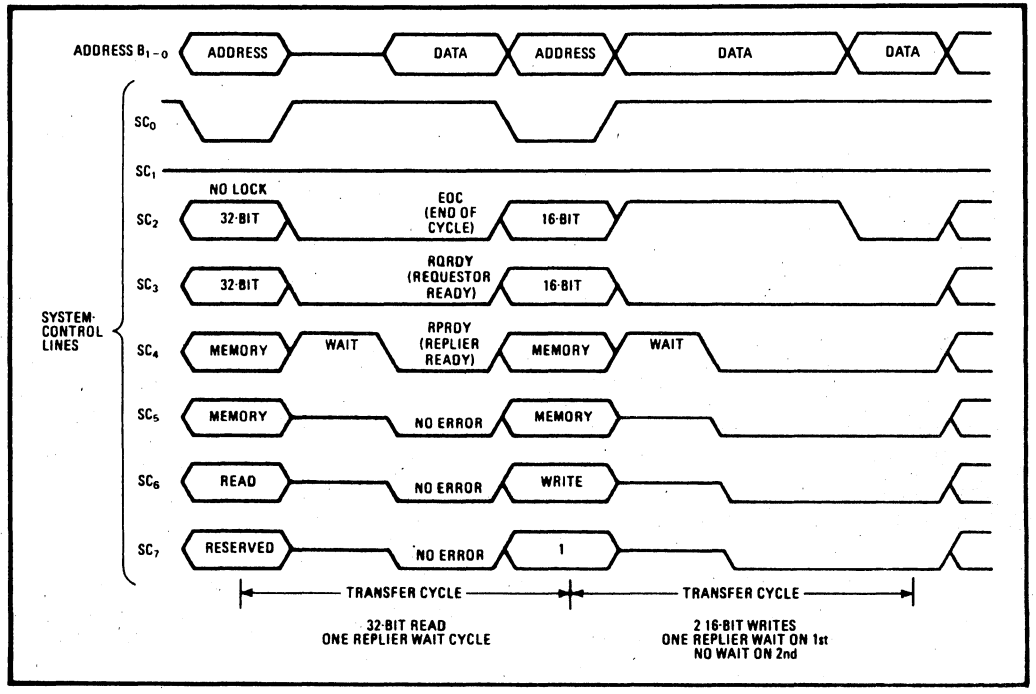
In this method, the modules exchange a copy of the data structure, which prevents the recipient module from corrupting the original data if something goes wrong. The major problem is performance—the processors must execute code to move the data.

Multibus II's message passing combines the best of both methods and gives it hardware support. Message passing uses a hardware pass-by-value interface that gives the performance of a pass-by-reference system. In addition, the software module used by both methods are replaced with a specialized message-passing interface. The processor passes-by-reference the reference to the data structure to the message-passing interface. This interface communicates with the destination module's interface to pass-by-value the data without processor intervention and is performed in the message-address space (Fig. 2). In many ways, it is helpful to think of the two communicating message-passing interfaces as a distributed intelligent DMA controller.

Combining methods has several benefits. First, the message-passing interfaces can take advantage of the bus's full capabilities, independent of the type or nature of the controlling processors. Even 8-bit processors or I/O boards can use all the bus's capabilities. As a result, significantly higher intermodule-communication performance is possible than with a software-based method. Another benefit is the elimination of shared memory: neither dual-ported memory structures nor global memory boards are needed. All data transfers take place in the logical message-address space rather than in the conventional physical-memory space.

The primary benefit of message passing is the uniform software interface. Hardware may be replaced with new modules that implement newer technology—such as moving from a single-density disk controller to a double-density version—without making any changes to driver software. Multibus II's architecture is defined to accommodate a wide cost-performance spectrum. At the high end of performance, its multiple-bus structures and 32-bit data paths let it reach system requirements far beyond those allowed by traditional bus architectures. On the low-cost side, the architecture accommodates the iSSB and a versatile form factor.

The increasing functionality of very large-scale integra-



4. Transfer cycle. Synchronous clocking is used in a typical data-transfer cycle; all lines are sampled on the bus clock's edge. The first of the two transfer cycles shown is a 4-byte memory-read cycle; the second is a burst-write cycle with two 16-bit data transfers.

tion has driven the price-performance ratio under 1:1 for a given board size while maintaining constant costs. In addition, the iSSB lets very large-scale integration drive the costs down while maintaining roughly constant functionality. The bus does this by reducing interconnection costs and allowing the physical distribution of modules.

A factor that limits system cost is the module interconnection. Parallel buses, particularly high-performance ones, require relatively expensive backplanes and multiple-line bus interfaces. The iSSB, basically a serial version of the iPSB, defines a message-passing interface identical to that used on the iPSB—yet it requires only a two-line interconnection. It does not need a backplane; a simple twisted pair of wires will do.

The iSSB has a maximum of 32 nodes distributed over a maximum of 10 meters of cable. The nodes may be distributed along an external cable segment or clustered into backplanes (Fig. 3). Each backplane may have 20 nodes, the maximum number of card slots in an iPSB backplane. Cluster nodes use repeaters to connect the backplanes and the iSSB cable. The repeaters isolate the cable from the excessive backplane capacitive load.

The iSSB uses an access protocol based on carrier-sense multiple-access with collision detection and deterministic delivery. When a node has a message to transmit, it looks at the bus for traffic before beginning transmission. If the bus is busy, the node waits until the line becomes idle and until an interface space has passed. Then it begins transmission.

When several nodes simultaneously initiate a transmission, a collision occurs. The CSMA/CD protocol handles these collisions through a deterministic collision-resolution algorithm that uses the principle of time slotting. This algorithm guarantees a time slot for each node during which it can transmit without interference from other nodes. This type of collision resolution guarantees a real-time response so that nodes can resolve collisions in a finite time. The iSSB uses a 16-bit CRC to detect transmission errors. The intelligent message-passing iSSB interface generates and examines the CRCs, making the iSSB as reliable as the iPSB.

The iSSB's physical interface consists of two signal lines that are included as part of the iPSB backplane design. Nodes encode data on the complementary open-collector signal lines (Table 2). The portion of the signal lines within the backplane is designed to operate in a high-noise environment, such as a heavily loaded backplane, relative to a coaxial cable. Cable extensions adhere

to standard transmission-line requirements.

The other feature that helps the Multibus II architecture address low-cost applications is its form factor. Multibus II boards follow the Eurocard mechanical standard and thus take advantage of the modular-board sizing it offers by specifying two compatible form factors: 233 by 220 millimeters and 100 by 220 mm. Because of the Eurocard standard's mechanical structure, it is possible for both sizes to coexist in the same card cage. Multibus II systems may be built totally with either the smaller or the larger board sizes, or the two sizes may be mixed in the same cage.

In a typical data-transfer cycle (Fig. 4), the bus uses synchronous clocking, so all lines are sampled on the edge of the bus clock. The first of the two transfer cycles is a simple 4-byte memory-read cycle; the second shows a burst-write cycle with two 16-bit data transfers.

A transfer cycle starts with the request phase. In this phase, the bus owner places address and command information on the bus. This information defines the replying agent and the type of operation, as well as the operation's address-space selection. The request phase lasts one clock cycle and is indicated by the system-control line SC_0 being active.

There are 10 system-control lines (Table 3). SC_0 is the line indicating that the request phase is active. The lock line, or SC_1, is used to lock out other parts of multiple-ported resources during indivisible operations such as test-and-set for semaphores. The next two lines (SC_2 and SC_3) encode the transfer's width.

One of the four Multibus II address spaces (memory, I/O, interconnect, or message) is selected through SC_4 and SC_5. SC_6 selects either a read or a write operation, and SC_7 is reserved. The system-control lines are parity-protected using a 4-bit parity on the SC_8 and SC_9 lines.

The user requesting information drives all system-control lines during the request phase. The reply phase starts immediately following the single clock-request phase; it lasts until the request and replying agents can exchange data, provided no time-outs occur. This exchange takes place using a two-sided handshake: both requestor and replier must agree that the data is valid.

During the reply phase, most system-control lines are redefined. SC_0 and SC_1 retain their request-phase meanings. SC_2 becomes the end-of-cycle control line. In this example, the requesting agent activates the EOC line to signal that the next data transfer is the last one. Because the first transfer cycle in the example contains only one data cycle, the EOC line is active throughout the reply phase. The next two lines form the two-sided handshake: REQRDY (requestor ready) and REPRDY (replier ready). The requestor can take data immediately, while the replier forces one wait cycle before it supplies valid data on the address-data lines.

The replying agent gets the opportunity to give the requesting agent additional status information using the SC_5 through SC_7 agent-error lines. These lines encode seven operational errors, like attempting to write to read-only resources or attempting a 32-bit access from an 8-bit one. These errors may induce retry operations by an intelligent bus interface or may be passed to the offending processor as error interrupts.

The remaining two lines are for the corresponding 4-bit parity. For the reply phase, the requesting agent drives the SC_0 through SC_3 lines and their SC_9 parity bit, while the replying agent drives SC_4 through SC_7 and their SC_8 parity bit. The transfer cycle terminates when EOC, REQRDY, and REPRDY are all active.

The second transfer cycle exhibits a two data-cycle burst write operation to memory. The replying agent injects one wait cycle before accepting the first 16 bits of data, although it takes the second 16 bits on the following clock. A burst transfer may be unlimited in length, although most implementations will restrict it to some maximum—usually 16 to 32 bytes—to ensure that no agent consumes the entire bus bandwidth. □

TABLE 3: SYSTEM CONTROL LINE ENCODINGS

System-control-line no.	Request phase	Reply phase
SC_0	low	high
SC_1	lock	lock
SC_2	data width	EOC
SC_3	data width	REQRDY
SC_4	address space	REPRDY
SC_5	address space	agent error
SC_6	read/write	agent error
SC_7	reserved	agent error
SC_8	parity SC_{4-7}	parity SC_{4-7}
SC_9	parity SC_{0-3}	parity SC_{0-3}

SC_3	SC_2	Data width	SC_5	SC_4	Address space
High	H	8 bits	H	H	memory
H	Low	16 bits	H	L	input/output
L	H	24 bits	L	H	message
L	L	32 bits	L	L	interconnect

SC_7	SC_6	SC_5	Agent error
H	H	H	no error
L	H	H	negative acknowledge (message)
H	L	H	continuation error
L	L	H	reserved
H	H	L	width error
L	H	L	reserved
H	L	L	not understood
L	L	L	reserved

intel

ARTICLE REPRINT

AR-350

October 1984

Message Passing Supports Multiple Processor Design

STEPHEN J. PACKER AND NARJALA BHASKER

Order Number: 280067-001

MESSAGE PASSING SUPPORTS MULTIPLE PROCESSOR DESIGN

Enhancing message-passing capabilities on Multibus II allows efficient data transmission among multiple processors.

by Stephen J. Packer and Narjala Bhasker

As microcomputer systems have evolved into complex multiple processor designs, they have been very difficult to build. This is because there has been no adequate solution to the problem of interprocessor cooperation. Until recently, the system programmer has been forced to provide software algorithms that are either very complex and slow, or are not extensible to more than two processors. Now, however, the message passing facility of Multibus II provides a hardware solution for interprocessor communication. At the same time, it gives the system programmer a standard software interface well suited for creating a distributed microcomputer-based operating system. Moreover, a special message space provision allows users to name modules with a message address rather than taking up memory address space.

Early microcomputers based on a single microprocessor depended on various support chips for such functions as I/O and memory access. The integration level of these support functions required more than one PC board for use with reasonably sized systems. The Multibus I system architecture allowed separate boards supporting various microcomputer functions to be interconnected. To minimize the cost of the support circuitry needed to interface the local bus to the external bus, the local bus architecture of the chosen processor was extended. Thus, Multibus I supported the memory, I/O address space, and bit transfer width of 8- and 16-bit microprocessors.

The bus I/O space was used to access various I/O devices such as the serial universal asynchronous receiver/transmitter (UART). Even complex I/O boards for disk and tape controllers used the I/O space for control. It soon became apparent, however, that a single processor could not handle the I/O functions required for appropriate performance.

Dividing the system into specialized functions

Intelligent I/O controllers, devised to off-load many complex I/O functions, leave more of the processor bandwidth for the application. This partitioning of the system into specialized functions, usually requiring an entire board, is quite natural. Using shared memory for data exchange between the

Stephen J. Packer is manager of Multibus II serial architecture at Intel Corp, 5200 NE Elam Young Pkwy, Hillsboro, OR 97123, where he is responsible for Multibus II message passing design and implementation. He holds a BS and an MS in electrical engineering from the University of Washington.

Narjala Bhasker is a software engineer at Intel Corp, Hillsboro, OR, where he is responsible for message passing on the Multibus II serial system bus. He holds a BS in electronic engineering from the Indian Institute of Technology, Madras, India and an MS in computer science from Syracuse University.

application processor and the special purpose, functionally partitioned modules is also natural.

In today's microcomputer systems, the system bus that interconnects functional modules is used almost exclusively for data movement between modules rather than for program execution. This data is encapsulated in control structures and is considered an interprocessor message. Memory-mapped control structures of interprocessor messages have variations as numerous as the programmers who program them. Only in those cases where hardware interfaces exist has any controlling standard emerged. The

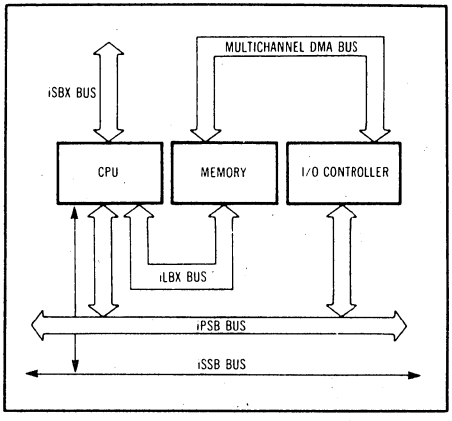
Extending Multibus I into Multibus II

To provide computational power for increasingly complex applications, microcomputer systems have evolved from basic single-processor systems to more intricate multiple processor systems that distribute the total processing load among various hardware modules. Intel's Multibus I system architecture developed a multiple bus structure approach for these complex multiple processor systems. Using a design strategy known as functional partitioning, the Multibus I architecture provided a specialized bus for specific critical functions, thus preserving the bandwidth of the system bus for interprocessor communication and data movement.

The Multibus II system architecture refines this approach and extends its range (see Table). By providing a traditional microprocessor bus for the access of memory and I/O address space via the parallel system bus (the iPSB), Multibus II continues the evolutionary path established by Multibus I. It also prepares the way for future 32-bit microcomputers by providing 32-bit data and address paths to memory. The bus clock frequency of 10 MHz provides a maximum 40-Mbyte/s transfer rate that anticipates performance requirements for future microprocessors.

Like Multibus I, Multibus II permits multiple bus masters capable of requesting and arbitrating for access to the bus. The Multibus II arbitration policy is more involved and allows complex algorithms as well as avoiding access starvation for all modules. In addition, Multibus II provides the centralization of bus functions in a single module. This reduces the cost of multiboard systems, since all boards need not carry the overhead of providing bus-level functions such as clock and timeout.

For designs using large amounts of RAM and executing from this memory with minimal delays (see Figure), the Multibus II specification provides a local bus extension (iLBX II). Multiple iLBX II execution buses can exist in a single Multibus II system that isolates the processors' execution environments from one another and leaves the system bus for data movement between environments.



The serial system bus (iSSB) is offered as the lowest cost method of interconnecting functional modules in a system. The processor interface is identical to the message-passing facility in the parallel system bus except for initialization and error management.

Multibus II Specifications in Brief

Parallel system bus (iPSB)

- 32-bit address and data path width
- 16-bit I/O address and data width
- synchronous operation with clock rate of 10 MHz
- 40-Mbyte/s transfer rate (sequential-transfer)
- up to 20 agents
- support of 8-, 16- and 32-bit processors
- 8-, 16-, 24-, 32-bit transfers
- no starvation arbitration policy
- central system functions; clock, timeout, power fail
- transfer parity, DIN connectors, distributed ground pins
- interconnect address space (512 eight-bit, one interrupt line registers) for system-level diagnostics and configuration
- 256 interrupt sources
- message passing

Local bus extension (iLBX II)

- 32-bit memory and 26-bit address path width
- 16-bit I/O address and data width
- synchronous operation with a clock rate of 12 MHz
- 48-Mbyte/s transfer rate (sequential-transfer)
- up to 6 agents (one master and one secondary master)
- pipelining (overlap of address and data cycles
- support of 8-, 16- and 32-bit processors
- 8-, 16-, 32-bit transfers
- optional transfer parity, DIN connectors
- one interrupt line

Serial system bus (iSSB)

- bus clock rate of 2 MHz
- up to 32 nodes on a maximum of 10 m of cable
- up to 20 nodes on one backplane CSMA/CD access method
- deterministic collision resolution
- 16-bit cyclic redundancy check

System bus extension (iSBX)

- low cost I/O extension to CPU board

Multichannel DMA bus

- 16-bit address and data path width
- asynchronous operation with clock rate of 2 MHz
- 8-, and 16-bit transfers
- up to 16 nodes on cable of up to 15

purpose of these standard interfaces has been to provide a software-compatible hardware upgrade to earlier products. Hardware available for message passing in an efficient and easy-to-use manner frees the system programmer for more important tasks. Any solution to the interprocessor communication problem comprises one of two options: a pass-by-reference interface, or a pass-by-value interface.

The pass-by-reference approach to interprocessor communication passes pointers between modules without copying the actual data. This requires a shared memory resource accessible by the cooperating processors (Fig 1). In tightly coupled systems having the same types of processors with shared memory and I/O, the pass-by-reference architecture is very effective.

With this method, however, there are often very serious impediments to microcomputer design. First, the hardware architecture of shared processors must be compatible, if not exactly the same, particularly if the processors are sharing executable code. This hinders the development of hybrid systems for processors with radically different internal structures.

A pass-by-reference architecture is very effective in tightly coupled systems with the same processor types.

For example, mixing 8-, 16-, and 32-bit processors with memory is extremely difficult if there are byte-alignment differences among the selected processors. Also, a processor with a wider data path than the accessed memory may be restricted to specific instructions that make only byte-data references. All too often, the software algorithms must be context sensitive to allow processor-independent implementations.

Achieving low cost and high performance execution requires that memory arrays be used on processor boards. For memory-mapped messages, this onboard memory must be dual-ported and accessible from the onboard CPU and the offboard modules via the Multibus interface. This forces the two processors to view the memory by different address ranges (Fig 2). The different addresses by which the processors know the same physical memory location are called aliases. Such "aliasing" results in a loss of performance since one of the processors must recalculate all the pointers to reach the same addresses.

More importantly, the software algorithms for managing shared data structures are not extensible to more than two processors without hardware help. When two-processor algorithms are used in a system with three or more processors, a shared data structure is needed for each communicating pair. Performance suffers in a server module because it must search a

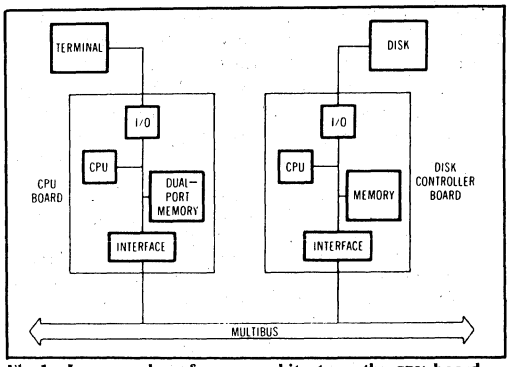


Fig 1 In a pass-by-reference architecture, the CPU board contains a microprocessor such as the Intel 80286, and supports logic, I/O devices, a dual-port RAM array, and a parallel system bus (iPSB) interface logic. The disk controller board has a processor such as the 80188, a parallel I/O interface, and sufficient RAM for sector caching. The disk controller communicates via messages contained in data structures in the dual-port RAM of the CPU board.

list of data structures for each requester. But, the most serious difficulty is configuring such a system when modules are optionally added or deleted.

Almost all memory-mapped, message-passing schemes assume a single-application processor and a dedicated slave processor. In future systems, the functional modules must become servers that will accommodate multiple application processors. In addition, the user must be able to add and remove application modules without disrupting the system. Finally, a pass-by-reference implementation does not lend itself to memory protected systems (like Microsoft's Xenix operating system) that are not object oriented. A failing processor or faulty software can easily compromise a system using shared memory.

Defining the pass-by-value interface

Another choice for a message-passing design is the pass-by-value strategy that copies data, rather than exchanging it, via pointers. This method is usually selected when protection criteria are more important

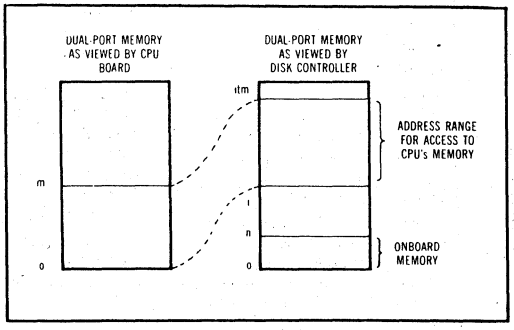


Fig 2 In the pass-by-reference scheme, the disk controller software views the CPU's memory in different address ranges. Aliasing complicates the software and severely reduces performance.

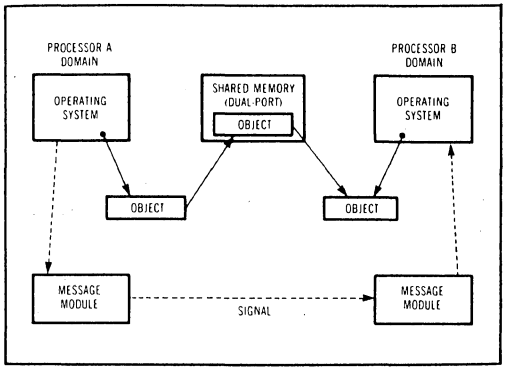


Fig 3 The operating system of processor A in the pass-by-value approach transfers a memory object to the operating system of processor B. It invokes a message-passing module that copies the object from private memory to a shared memory area. As processor B is interrupted, it calls out its message-passing module and copies the data to a private memory area.

than maximum system performance. Given the functional partitions of a distributed microcomputer system, a pass-by-value implementation requires a double copy of the data (Fig 3). The data is copied from application space to some internal system space, where it can be accessed by the second processor. After the first processor interrupts the second, the second can move the data to the memory space of the receiving application.

This process of double-copying the data places a considerable strain on the system's performance and yields a lower level of performance—the most serious disadvantage of the pass-by-value method. Nonetheless, a pass-by-value implementation is easier than a pass-by-reference method because differences in the hardware (eg, memory data width) are confined to the interface software. New modules are easier to develop since internal data structures need not be known to other modules. Finally, existing operating systems such as Xenix can be used without rewriting their internal structures.

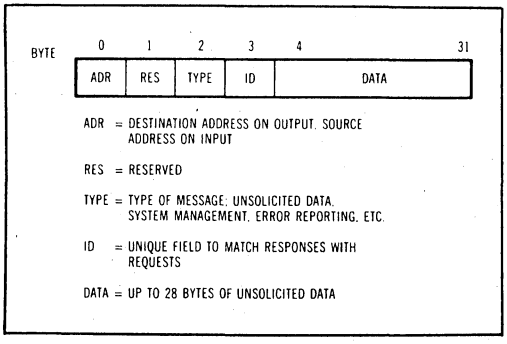


Fig 4 The message in an unsolicited data message format is from 4 to 32 bytes long in increments of 4 bytes. The type field indicates the exact function of the message, which may be an interrupt (with no data); a data message; or a local control message between the processor and the message device. The last category is not transmitted on the bus and hence is not specified in Multibus II.

An obvious choice for the system programmer is some high performance version of the pass-by-value method. Getting high performance, however, requires an architectural hardware change to augment the software algorithm. A distributed operating system and a distributed application have similar requirements for messages exchanged between modules. This is because both use independent processes that must communicate with other processes.

The messages received by the process providing a service to another process are unsolicited (ie, the receiver cannot predict when a message will occur). These unsolicited messages are used to negotiate the movement of varying and potentially large amounts of data. This partitioning of the kinds of messages expected in a system can be used to define a new model for message passing.

A special Multibus II message space provides a means of naming modules by a message address rather than by using a memory address. This message address space is much smaller than the address space of the supported processors, and the message-passing facility is independent of the physical medium implementation interconnecting the functional modules. This permits a software design that can support a wide range of systems—from those that use the low cost serial bus to the high performance parallel bus. This design will also eventually support the interconnection of single-chip functional modules.

Unsolicited messages can either be requests for service or a reply to a request for service (Fig 4). Thus, messages become the fundamental basis for requester/server implementations of distributed processes. The most important requirement for unsolicited messages is high efficiency and low latency for the movement between functional modules (ie, the cost of sending the message must be a fraction of the task switching time for the operating system). In addition, any delays must be short. Commands can therefore be thought of as processor interrupts with data.

Unsolicited messages generated in a system are usually quite short—usually less than 32 bytes. Nonetheless, they carry the needed control information for such diverse functions as global object management and I/O control. Within a single system, the total system capacity needed to generate unsolicited messages is self-limiting. This is due to a limit on the number of requests that can be sent before a reply is required to continue processing.

These two attributes make unsolicited messages ideally suited for a hardware first in, first out (FIFO) buffer (Fig 5). Messages bounded in size and number can be placed in the output FIFO of the sending module, and then removed by the receiving module. The FIFO implementation has two advantages. First,

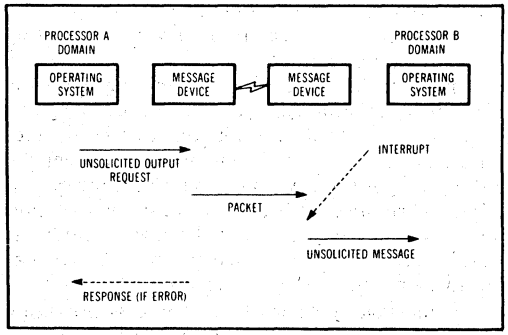


Fig 5 An unsolicited output message is moved into the message module first in, first out (FIFO). This is very efficient for processors with a string move instruction. The message devices cooperate to move a packet containing the unsolicitied output message to the receiving module FIFO. The receiving operating system can remove the message from the FIFO on interrupt with a string move or a byte-by-byte read of an I/O port.

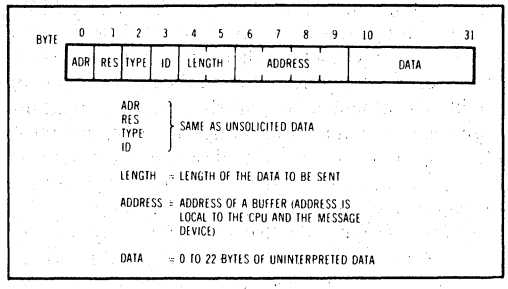


Fig 6 In a solicited output message format, two additional fields appear in the solicited message, indicating the address and the length of the data to be sent. Only the length field is actually transmitted on the bus because the address is local to the processor and its message device.

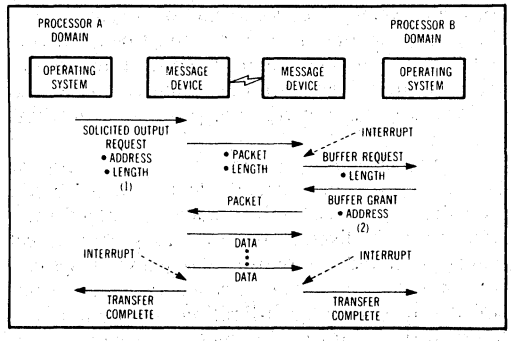


Fig 7 The data address in the solicited message procedure is retained in the sending message device and the length forwarded to the receiving device. On interrupt, the receiving operating system receives the buffer request and replies with a buffer grant message. The message devices move the data, up to 64 Kbytes, without further processor action. On completion, both sender and receiver get a transfer complete message.

it is easy to buffer messages that arrive with a random distribution, processed at a fixed rate. It is also easy to implement a "FIFO full" recovery strategy for those rare cases where messages are generated in excess of the chosen FIFO's capacity. Thus, overall system performance can be determined by the performance of the FIFO mechanism, not by the efficiency of the "FIFO full" recovery.

Second, movement of the message to the FIFO immediately frees the memory containing the message. This is a significant advantage to the operating system since a subsequent interrupt and context switch is not necessary in order to release the buffer at a later time.

When placed in FIFO, unsolicited messages are delivered to the operating system. This expectation is based on the high reliability of a computer bus where errors are very rare, and extraordinary recovery procedures can be tolerated. Thus, the inability to deliver a message is immediatly reported to the sender, and the recovery action determined by the appropriate software.

Processing solicited messages

Eventually, the exchange of unsolicited messages results in the need to move a large amount of data. The bulk data movement can be accomplished by many unsolicited messages, but the cost of processing interrupts in the receiving module would bog down the system. Furthermore, such interrupts would no longer arrive randomly and in the limited number originally assumed.

An acceptable alternative is a DMA facility between functional partitions in which each end independently authorizes transfers. This facility must not compromise the low latency required by unsolicited messages. The solicited message facility of Multibus II provides this feature.

Solicited messages are initiated with a special message placed in the FIFO. The solicited message contains specific fields that define the origin and length of a buffer in the sending system's local memory (Fig 6). The length field is sent to the receiving module FIFO, where it appears as a request to allocate a buffer of specific length.

After allocating the buffer, the receiving module sends a message through its output FIFO, carrying the address of its buffer. The data movement then takes place without any further involvement of the processor in the modules. Upon completion of the data transfer, messages are generated and placed in both the sender's and receiver's input FIFO which signals the completion and return status. At this point, ownership of the buffers returns to the modules (Fig 7).

There are several advantages to this method. First, the assignment of buffers is completely under the control of the functional modules that own them. Thus, this facility is compatible with the memory

protection features of the operating system. All pointers are used only in their natural address range. Second, the actual data movement can be controlled to optimize the bandwidth of the various buses in the system. This increases overall system performance. The data can be retrieved or stored from the module's memory at the rate of the internal bus on the module.

Pass-by-value message-passing is usually chosen when protection criteria are more important than maximum system performance.

Since the data transfer can be done at the full bandwidth of the interconnect bus, running all three buses at the speed of the slowest bus is not necessary. The module's CPU is not involved in this transfer except for potentially delayed access to its local memory bus during actual transfer.

The data is made into packets to suit the physical medium or to meet realtime needs for low latency access to the interconnect bus. The packets are transferred on the module interconnect bus at the optimum speed, whether serial or parallel, while unused bus bandwidth remains available to other communicating modules.

The design of the Multibus II message-passing facility can be implemented in a single VLSI device. The architecture offers a very simple device that is used only to support interrupts (unsolicited messages with no data). A more complex device is needed to support unsolicited messages with data and solicited messages. This device requires a FIFO controller, a DMA controller, a small amount of RAM for packet buffering, and a logic control unit. The serial system bus device is a derivative of the parallel message device with the replacement of the 32-bit bus interface with a serial interface unit.

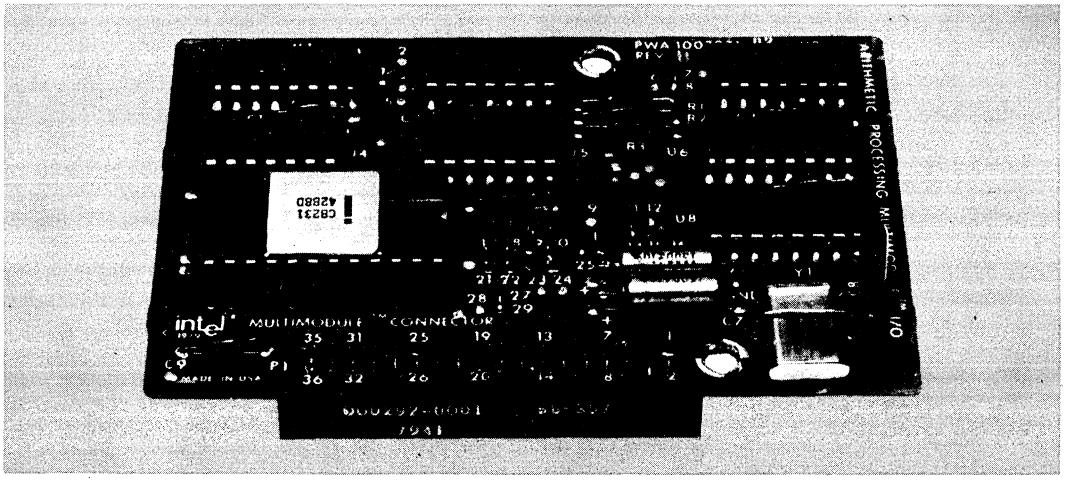
While Multibus II-compatible boards do not require custom VLSI, boards designed by Intel will support the message-passing facility of Multibus II when the VLSI devices are fully tested and qualified. These VLSI devices will be available to ensure the rapid acceptance of the Multibus II specification.

High Speed Math Boards 5

iSBX™ 331
FIXED/FLOATING POINT MATH MULTIMODULE BOARD

- **iSBX™ bus compatible high speed fixed/floating point math expansion**
- **4 MHz operation**
- **Fixed point single and double precision (16/32-bit)**
- **Floating point double precision (32-bit)**
- **Binary data formats**
- **Add, subtract, multiply and divide**
- **Trignometric and inverse trigonometric functions**
- **Square root, log, and exponential functions**
- **Float-to-fixed and fixed-to-float conversions**
- **End of operation interrupt**
- **Software reset control**
- **Low power requirements**
- **iSBX™ bus on-board expansion eliminates MULTIBUS® system bus latency and increases system throughput**

The Intel® iSBX 331 Fixed/Floating Point Math MULTIMODULE Board is a member of Intel's new line of iSBX bus compatible MULTIMODULE products. The iSBX MULTIMODULE board plugs directly into any iSBX bus compatible host board offering low cost incremental on-board expansion. As a result, any iSBX bus compatible host board may be expanded to perform high speed math computations, affording up to a 40× improvement in speed compared to software math. The iSBX 331 module performs single/double (16/32-bit) precision fixed point plus double (32-bit) precision floating point arithmetic operations. In addition, the module performs transcendental, data manipulation, and fixed to float/float to fixed point conversion operations. The command operations run entirely independent of the host board permitting efficient concurrent processing. The iSBX board is closely coupled to the host board through the iSBX bus, and as such, offers maximum on-board performance and frees MULTIBUS system traffic for other system resources. Incremental power dissipation is minimal requiring only 2.73 watts.



FUNCTIONAL DESCRIPTION

The iSBX 331 module uses the Intel 8231 Arithmetic Processing Unit (APU) to accomplish high speed (4 MHz) math operation. The system software may communicate with the iSBX 331 module across the iSBX bus using I/O read/write commands. All transfers, including operand, result, status, and command information, take place over an 8-bit bidirectional data bus. Operands are pushed onto an internal stack and commands are issued to perform operations on the data. Results are then available from the stack. A status byte may be read to monitor execution completion and the nature of the result (zero, sign, or errors). In addition, control logic is included on the iSBX 331 module to facilitate single instruction software reset control.

Command Functions

The iSBX 331 module commands fall into three categories: double precision floating point, single precision fixed point, and double precision fixed point (see Table 1). There are four arithmetic operations that can be performed in either fixed or floating point numbers: add, subtract, multiply, and divide. These operations require two operands. The 8231 assumes these operands are located in the internal stack as Top of Stack (TOS) and Next on Stack (NOS). The result will always be returned to TOS. There are four types of transcendental operations that can be performed in floating point numbers: trigonometric functions, logarithms, exponentials, and square roots. The results of these operations will be returned to TOS. There are four types of data manipulation operations that can be performed in either fixed or floating point numbers: sign change of TOS, exchange of TOS and NOS and copying or popping operands onto or off of TOS. Fixed to floating point conversion can be performed on floating point instructions and floating point to fixed point conversion can be performed on fixed point instructions.

The execution times of the commands are shown in Table 2.

Interrupt Requests

There is one interrupt line from the APU that may generate an interrupt request to the host: END (MINTRI). The END interrupt line is active upon command completion. The END signal is cleared by a reset or status register read.

Installation

The iSBX 331 module plugs directly into the female iSBX connector on the host board. The module is then secured at one additional point with nylon hardware to insure the mechanical security of the assembly (see Figures 1 and 2).

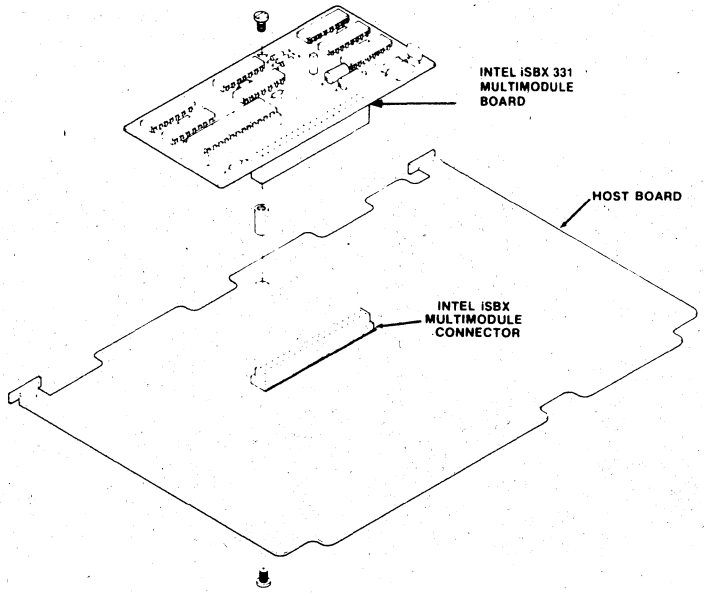


Figure 1. Installation of iSBX™ 331 Module on a Host Board

Table 1. Command Summary

Double Precision Floating Point Instructions (32-Bit)

Instruction	Description	Hex Code	Stack Contents After Execution[1] A B C D	Status Flags Affected[3]
ACOS	Inverse Cosine of A	0 6	R U U U	S, Z, E
ASIN	Inverse Sine of A	0 5	R U U U	S, Z, E
ATAN	Inverse Tangent of A	0 7	R B U U	S, Z
CHSF	Sign Change of A	1 5	R B C D	S, Z
COS	Cosine of A (radians)	0 3	R B U U	S, Z
EXP	e^A Function	0 A	R B U U	S, Z, E
FADD	Add A and B	1 0	R C D U	S, Z, E
FDIV	Divide B by A	1 3	R C D U	S, Z, E
FLTD	32-Bit Fixed to Floating Point Conversion	1 C	R B C U	S, Z
FLTS	16-Bit Fixed to Floating Point Conversion	1 D	R B C U	S, Z
FMUL	Multiply A and B	1 2	R C D U	S, Z, E
FSUB	Subtract A from B	1 1	R C D U	S, Z, E
LOG	Common Logarithm (base 10) of A	0 8	R B U U	S, Z, E
LN	Natural Logarithm of A	0 9	R B U U	S, Z, E
POPF	Stack Pop	1 8	B C D A	S, Z
PTOF	Stack Push	1 7	A A B C	S, Z
PUPI	Push π onto Stack	1 A	R A B C	S, Z
PWR	B^A Power Function	0 B	R C U U	S, Z, E
SIN	Sine of A (radians)	0 2	R B U U	S, Z
SQRT	Square Root of A	0 1	R B C U	S, Z, E
TAN	Tangent of A (radians)	0 4	R B U U	S, Z, E
XCHF	Exchange A and B	1 9	B A C D	S, Z

Double Precision Fixed Point Instructions (32-Bit)

Instruction	Description	Hex Code	Stack Contents After Execution[1] A B C D	Status Flags Affected[3]
CHSD	Sign Change of A	3 4	R B C D	S, Z, O
DADD	Add A and B	2 C	R C D A	S, Z, C, E
DDIV	Divide B by A	2 F	R C D U	S, Z, E
DMUL	Multiply A and B (R = lower 32 bits)	2 E	R C D U	S, Z, O
DMUU	Multiply A and B (R = upper 32 bits)	3 6	R C D U	S, Z, O
DSUB	Subtract A from B	2 D	R C D A	S, Z, C, O
FIXD	Floating to Fixed Point Conversion	1 E	R B C U	S, Z, O
POPD	Stack Pop	3 8	B C D A	S, Z
PTOD	Stack Push	3 7	A A B C	S, Z
XCHD	Exchange A and B	3 9	B A C D	S, Z

Table 1. Command Summary (continued)

Single Precision Fixed Point Instructions (16-Bit)

Instruction	Description	Hex Code	Stack Contents After Execution[2] A_U A_L B_U B_L C_U C_L D_U D_L	Status Flags Affected[3]
CHSS	Change Sign of A_U	7 4	R A_L B_U B_L C_U C_L D_U D_L	S, Z, O
FIXS	Floating to Fixed Point Conversion	1 F	R B_U B_L C_U C_L U U U	S, Z, O
POPS	Stack Pop	7 8	A_L B_U B_L C_U C_L D_U D_L A_U	S, Z
PTOS	Stack Push	7 7	A_U A_U A_L B_U B_L C_U C_L D_U	S, Z
SADD	Add A_U and A_L	6 C	R B_U B_L C_U C_L D_U D_L A_U	S, Z, C, E
SDIV	Divide A_L by A_U	6 F	R B_U B_L C_U C_L D_U D_L U	S, Z, E
SMUL	Multiply A_L by A_U (R = lower 16 bits)	6 E	R B_U B_L C_U C_L D_U D_L U	S, Z, E
SMUU	Multiply A_L by A_U (R = upper 16 bits)	7 6	R B_U B_L C_U C_L D_U D_L U	S, Z, E
SSUB	Subtract A_U from A_L	6 D	R B_U B_L C_U C_L D_U D_L A_U	S, Z, C, E
XCHS	Exchange A_U and A_L	7 9	A_L A_U B_U B_L C_U C_L D_U D_L	S, Z
NOP	No Operation	0 0	A_U A_L B_U B_L C_U C_L D_U D_L	

NOTES:

1. The stack initially is composed of four 32-bit numbers (A, B, C, D). A is equivalent to Top Of Stack (TOS) and B is Next On Stack (NOS). Upon completion of a command the stack is composed of: the result (R); undefined (U); or the initial contents (A, B, C, or D).
2. The stack initially is composed of eight 16-bit numbers (A_U, A_L, B_U, B_L, C_U, C_L, D_U, D_L). A_U is the TOS and A_L is NOS. Upon completion of a command the stack is composed of: the result (R); undefined (U); or the initial contents (A_U, A_L, B_U, B_L,...).
3. Nomenclature: Sign (S); Zero (Z); Overflow (O); Carry (C); Error Code Field (E).

Table 2. Command Execution Times

Command Mnemonic	μSeconds	Command Mnemonic	μSeconds
SADD	4.25	ASIN	1917
SSUB	7.5	ACOS	1933.5
SMUL	21-23.5	ATAN	1501.5
SMUU	20-24.5	LOG	1118.5-1783
SDIV	21-23.5	LN	1074.5-1739
DADD	5.25	EXP	948.5-1219.5
DSUB	9.5	PWR	2072.5-3008
DMUL	48.5-52.5	NOP	1
DMUU	45.5-54.5	CHSS	5.75
DDIV	52	CHSD	6.75
FIXS	23-54	CHSF	4.5
FIXD	25-86.5	PTOS	4
FLTS	24.5-46.5	PTOD	5
FLTD	24.5-94.5	PTOF	5
FADD	13.5-92	POPS	2.5
FSUB	17.5-92.5	POPD	3
FMUL	36.5-42	POPF	3
FDIV	38.5-46	XCHS	4.5
SQRT	200	XCHD	6.5
SIN	1116	XCHF	6.5
COS	1029.5	PUPI	4
TAN	1438.5		

NOTE: Assumes 4 MHz operation.

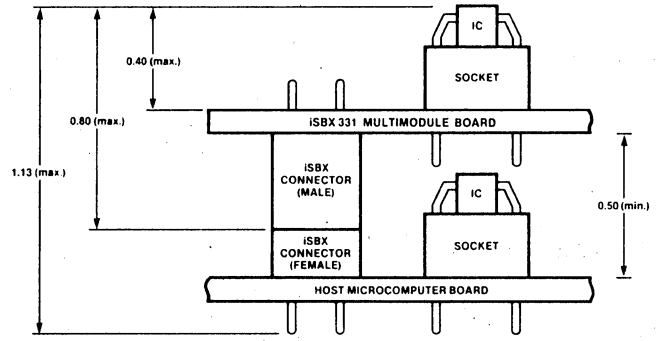


Figure 2. ISBX 331 MULTIMODULE Board Mounting Clearances (inches)

SPECIFICATIONS

Word Size

Data—8 bits.

On-Board Clock Rate

4.0 MHz ±0.1%.

I/O Addressing

Function	Type of Operation	iSBX Connector Port Address
Data Transfer	Read or Write	X0, X2, X4, or X6
Command Transfer	Write	X1, X3, X5, or X7
Status Transfer	Read	X1, X3, X5, or X7
Reset	Write	X8 through XF

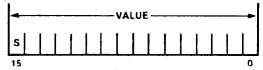
NOTE:
The port addresses are determined on the host iSBC microcomputer. Refer to the Hardware Reference Manual for your host iSBC microcomputer to determine the first digit (X) of the connector port addresses.

Arithmetic Functions

See Table 1.

Data Formats

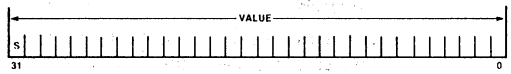
Single Precision Fixed Point (16 bits)



Bit 15: S = Sign of the operand. Positive values are represented by a sign bit of zero (S = 0). Negative values are represented by the two's complement of the corresponding positive value with a sign bit equal to 1 (S = 1).

Bits 0-14: Values in the range from −32, 768 to +32, 767.

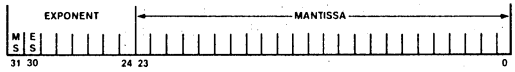
Double Precision Fixed Point (32 bits)



Bit 31: S = Sign of operand. Positive values are represented by a sign of zero (S = 0). Negative values are represented by the two's complemet of the corresponding positive value with a sign bit equal to 1 (S = 1).

Bits 0-30: Values in the range from −2, 147, 483, 648 to +2, 147, 483, 647.

Double Precision Floating Point (32 bits)



Bit 31: MS = Sign of the mantissa. 1 represents negative and 0 represents positive.

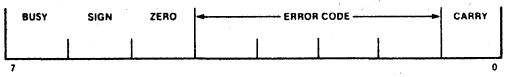
Bits 24-30: ES = the exponent expressed as a two's complement 7-bit value having a range of −64 to +63.

Bits 0-23: The mantissa is expressed as a 24-bit (fractional) value. The 8231 APU requires that floating point data be represented by a fractional mantissa value between 0.5 and 1 multiplied by 2 raised to an appropriate power (exponent). This is expressed as follows:

$$\text{Value} = \text{mantissa} \times 2^{\text{exponent}}$$

Device Status

Device status is provided by means of an internal status register whose format is shown below:



BUSY: Indicates that 8231 is currently executing a command (1 = Busy)

SIGN: Indicates that the value on the top of stack is negative (1 = Negative)

ZERO: Indicates that the value on the top of stack is zero (1 = Value is zero)

ERROR CODE: This field contains an indication of the validity of the result of the last operation. The error codes are:

0000 — No error
1000 — Divide by zero
0100 — Square root or log of negative number
1100 — Argument of inverse sine, cosine, or e^x too large
XX10— Underflow
XX01— Overflow

CARRY: Previous operation resulted in carry or borrow from most significant bit. (1 = Carry/Borrow, 0 = No Carry/No Borrow.)

If the BUSY bit in the status register is a one, the other status bits are not defined; if zero, indicating not busy, the operation is complete and the other status bits are defined as given above.

Access Time

Read—1900 ns (max.)
Write—1900 ns (max.)

NOTE:
Actual transfer speed is dependent upon the cycle time of the host microcomputer. The listed times assume no operation in progress. If an operation is executing when an access is attempted, the command execution time must be added to the above times for all accesses except status read.

Interrupts

One interrupt request may originate from the APU indicating command completion (END).

Interface

iSBX Bus—All signals TTL compatible

Physical Characteristics

Width—6.35 cm (2.50 in.)
Length—9.40 cm (3.70 in.)
Height*—2.04 cm (0.80 in.) iSBX 331 Board
—2.86 cm (1.13 in.) iSBX 331 Board + Host Board
Weight—51 gm (1.79 oz)

*See Figure 2.

Electrical Characteristics

DC Power Requirements

$V_{CC} = +5V \pm 5\%$ $I_{CC} = 365$ mA max.
$V_{DD} = +12V \pm 5\%$ $I_{DD} = 75$ mA max.

Environmental

Operating Temperature—0°C to 55°C

Free moving air across the base board and iSBX board.

Reference Manual

142668-01—iSBX 331 Floating Point Math MULTIMODULE Board (NOT SUPPLIED)

Reference manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

ORDERING INFORMATION

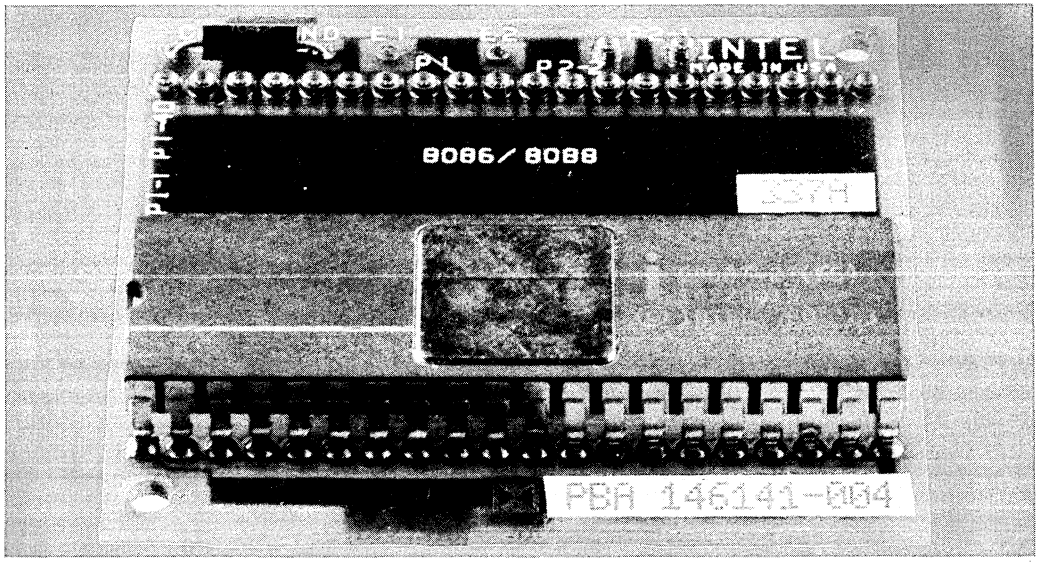
Part Number	Description
SBX 331	Fixed/Floating Point Math MULTIMODULE Board

iSBC® 337A and iSBC® 337 MULTIMODULE™ NUMERIC DATA PROCESSOR

- **High speed fixed and floating point functions for 8 or 5 MHz iSBC® 86, 88, and iAPX 86, 88 systems**
- **Extends host CPU instruction set with arithmetic, logarithmic, transcendental and trigonometric instructions**
- **MULTIMODULE™ option containing 8087 Numeric Data Processor**
- **Up to 80X performance improvement in Whetstone benchmarks over 8MHz iAPX-86/10 performance**
- **Supports seven data types including single and double precision integer and floating point**
- **Software support through ASM 86/88 Assembly Language and High Level Languages**
- **Fully supported in the multi-tasking environment of the iRMX™ 86 Operating System**

The Intel iSBC® 337A/337 MULTIMODULE™ Numeric Data Processor offers high performance numerics support for iSBC 86 and iSBC 88 Single Board Computer users, for applications including simulation, instrument automation, graphics, signal processing and business systems.

The coprocessor interface between the 8087 Numeric Data Processor and the host CPU provides a simple means of extending the instruction set with over 60 additional numeric instructions supporting seven data types. The MULTIMODULE implementation allows the iSBC 337A/337 module to be used on all iSBC 86/88" single board computers and can be added as an option to custom iAPX board designs.



October, 1984
ORDER NUMBER: 280077-001

OVERVIEW

The iSBC 337A/337 MULTIMODULE Numeric Data Processor (also called NDP) provides arithmetic and logical instruction extensions to the 86/88 of the iAPX 86/88 families. The instruction set consists of arithmetic, transcendental, logical, trigonometric and exponential instructions which can all operate on seven different data types. The data types are 16, 32, and 64 bit integer, 32 and 64 bit floating point, 18 digit packed BCD and 80 bit temporary.

Coprocessor Interface

The coprocessor interface between the host CPU and the iSBC 337A/337 processor provides easy to use and high performance math processing. Installation of the iSBC 337A/337 processor is simply a matter of removing the host CPU from its socket, installing the iSBC 337A/337 processor into the host's CPU socket, and reinstalling the host CPU chip into the socket provided for it on the iSBC 337A/337 processor (see Figure 1).

All synchronization and timing signals are provided via the coprocessor interface with the host CPU. The two processors also share a common address/data bus. (See Figure 2). The NDP component is capable of recognizing and executing NDP numeric instructions as they are fetched by the host CPU. This interface allows concurrent processing by the host CPU and the NDP. It also allows NDP and host CPU instructions to be intermixed in any fashion to provide the maximum overlapped operation and the highest aggregate performance.

High Performance and Accuracy

The 80-bit wide internal registers and data paths contribute significantly to high performance and minimize the execution time difference between single and double precision floating point formats. This 80-bit architecture provides very high resolution and accuracy.

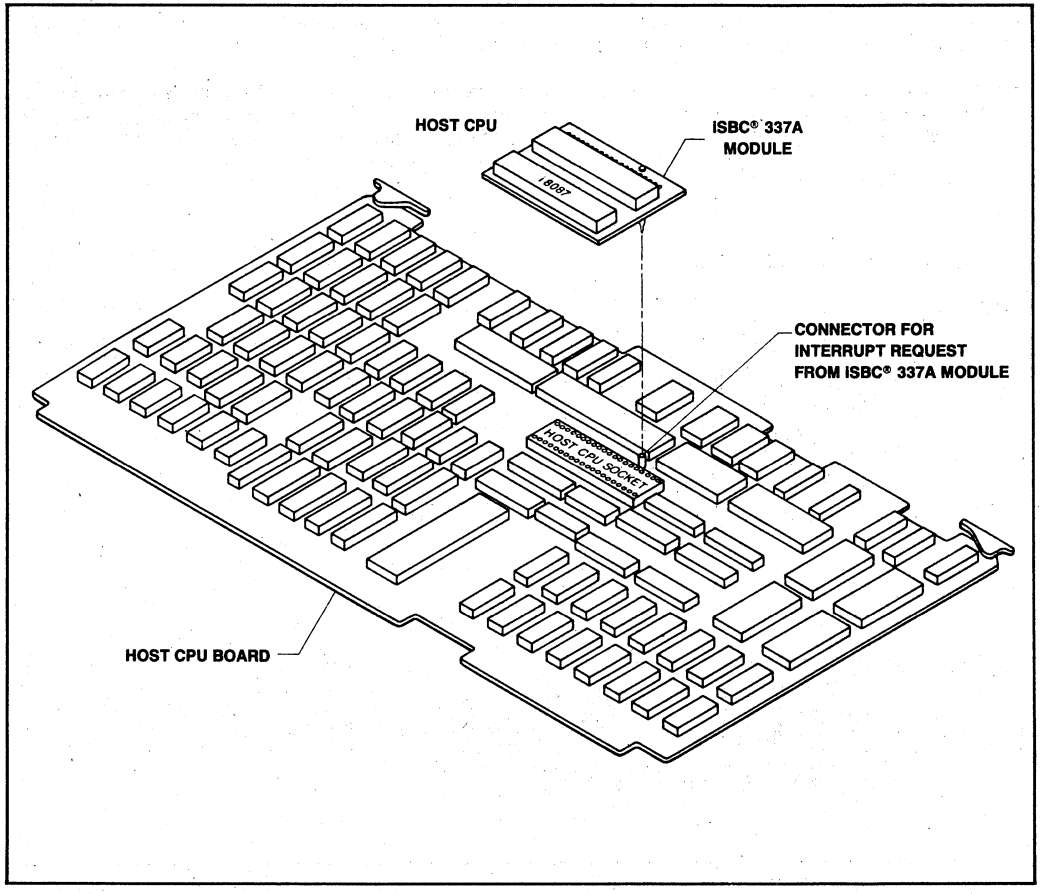


Figure 1. iSBC® 337A Module Installation

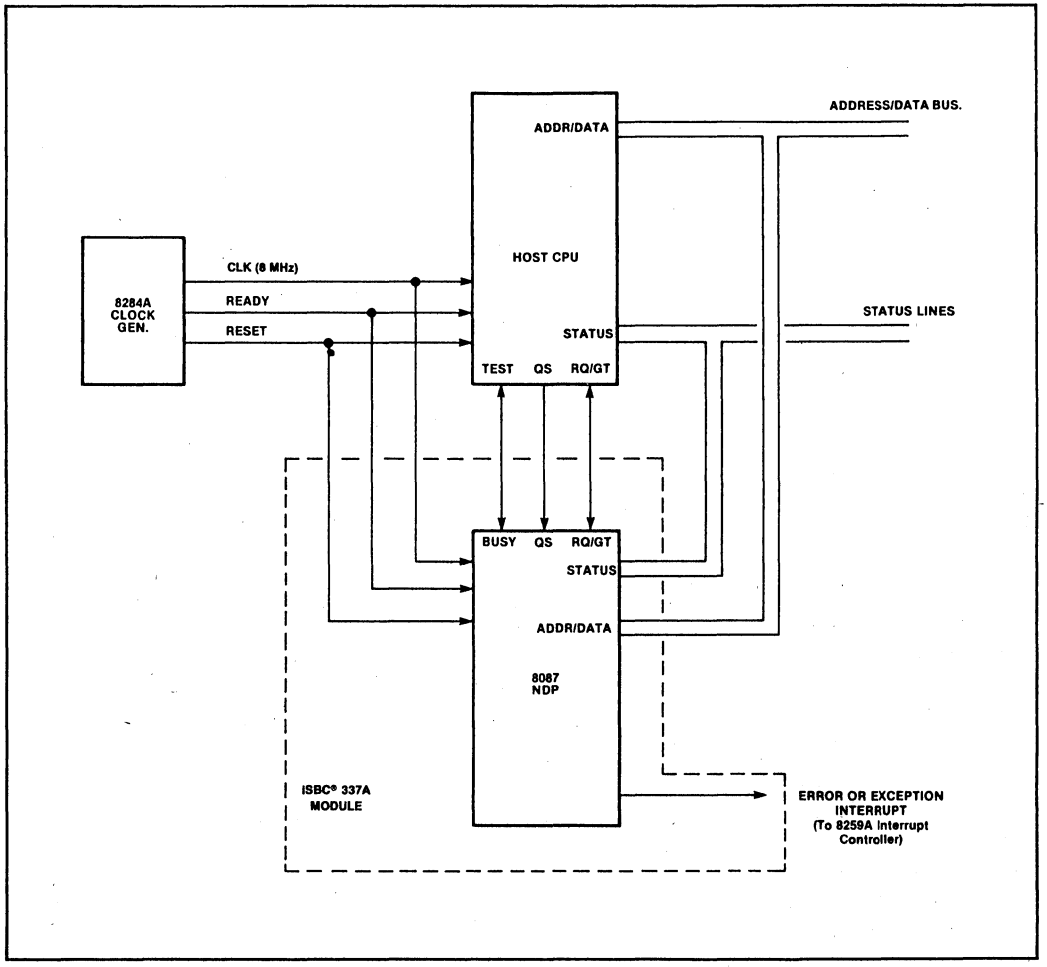


Figure 2. iSBC® 337A System Configuration

This precision is complemented by extensive exception detection and handling. Six different types of exceptions can be reported and handled by the NDP. The user also has control over internal precision, infinity control and rounding control.

SYSTEM CONFIGURATION

As a coprocessor to the Host CPU, the NDP is wired in parallel with the CPU as shown in Figure 2. The CPU's status and queue status lines enable the NDP to monitor and decode instructions in synchronization with the CPU and without any CPU overhead. Once started, the NDP can process in parallel with and independent of the host CPU. For resynchronization, the NDP's BUSY signal informs the CPU that the NDP is executing an instruction and the CPU WAIT instruction tests this signal to insure that the NDP is ready to execute subsequent instructions.

The NDP can interrupt the CPU when it detects an error or exception. The interrupt request line is routed to the CPU through an 8259A Programmable Interrupt Controller. This interrupt request signal is brought down from the iSBC 337A/337 module to the single board computer through a single pin connector (see Figure 1). The signal is then routed to the interrupt matrix for jumper connection to the 8259A Interrupt Controller. Other iAPX designs may use a similar arrangement, or by masking off the CPU "READ" pin from the iSBC 337A/337 socket, provisions are made to allow the now vacated pin of the host's CPU socket to be used to bring down the interrupt request signal for connection to the base board and then to the 8259A. Another alternative is to use a wire to establish this connection.

PROGRAMMABLE INTERFACE

Table 1 lists the seven data types the NDP supports and presents the format for each type. Internally, the NDP holds all numbers in the temporary real format. Load and store instructions automatically convert operands represented in memory as 16-, 32-, or 64-bit integers, 32- or 64-bit floating point numbers or 18-digit packed BCD numbers into temporary real format and vice versa.

Computations in the NDP use the processor's register stack. These eight 80-bit registers provide the equivalent capacity of 40 16-bit registers. The NDP register set can be accessed as a stack, with instructions operating on the top stack element, or as a fixed register set with instructions operating on explicitly designated registers.

Table 2 lists the NDP instructions by class. Assembly language programs are written in ASM 86/88, the iAPX family assembly language.

Table 3 gives the execution times of some typical numeric instructions and their equivalent time on a 8 MHz 8086-2.

FUNCTIONAL DESCRIPTION

The NDP is internally divided into two processing elements, the control unit (CU) and the numeric execution unit (NEU), providing concurrent operation of the two units. The NEU executes all numeric instructions, while the CU receives and decodes instructions, reads and writes memory operands and executes processor control instructions.

Control Unit

The CU keeps the NDP operating in synchronization with its host CPU. NDP instructions are intermixed with CPU instructions in a single instruction stream. The CPU fetches all instructions from memory; by monitoring the status signals emitted by the CPU, the NDP control unit determines when an 8086-2 instruction is being fetched. The CU taps the bus in parallel with the CPU and obtains that portion of the data stream.

After decoding the instruction, the host executes all opcodes but ESCAPE (ESC), while the NDP executes only the ESCAPE class instructions. (The first five bits of all ESCAPE instructions are identical). The CPU does provide addressing for ESC instructions, however.

Table 1. 8087 Datatypes

Data Formats	Range	Precision	Most Significant Byte (7 07 07 07 07 07 07 07 07 07 0)
Word Integer	10^4	16 Bits	I_{15} ... I_0 — Two's Complement
Short Integer	10^9	32 Bits	I_{31} ... I_0 — Two's Complement
Long Integer	10^{19}	64 Bits	I_{63} ... I_0 — Two's Complement
Packed BCD	10^{18}	18 Digits	S — D_{17} D_{16} ... D_1 D_0
Short Real	$10^{\pm 38}$	24 Bits	S E_7 E_0 F_1 ... F_{23} — F_0 Implicit
Long Real	$10^{\pm 308}$	53 Bits	S E_{10} E_0 F_1 ... F_{52} — F_0 Implicit
Temporary Real	$10^{\pm 4932}$	64 Bits	S E_{14} E_0 F_0 ... F_{63}

Note:

Integer: I **Sign: S**
Fraction: F **BCD Digit (4 Bits): D**
Exponent: E

Packed BCD: $(-1)^S(D_{17} \ldots D_0)$

Real: $(-1)^S(2^{E-BIAS})(F_0F_1 \ldots)$

Bias = 127 for Short Real
1023 for Long Real
16383 for Temp Real

Table 2. 8087 Instruction Set

Data Transfer Instructions

Real Transfers	
FLD	Load real
FST	Store real
FSTP	Store real and pop
FXCH	Exchange registers
Integer Transfers	
FILD	Integer load
FIST	Integer store
FISTP	Integer store and pop
Packed Decimal Transfers	
FBLD	Packed decimal (BCD) load
FBSTP	Packed decimal (BCD) store and pop

Comparison Instructions

FCOM	Compare real
FCOMP	Compare real and pop
FCOMPP	Compare real and pop twice
FICOM	Integer compare
FICOMP	Integer compare and pop
FTST	Test
FXAM	Examine

Transcendental Instructions

FPTAN	Partial tangent
FPATAN	Partial arctangent
F2XM1	2^X-1
FYL2X	$Y \cdot \log_2 X$
FYL2XP1	$Y \cdot \log_2(X+1)$

Arithmetic Instructions

Addition	
FADD	Add real
FADDP	Add real and pop
FIADD	Integer add
Subtraction	
FSUB	Subtract real
FSUBP	Subtract real and pop
FISUB	Integer subtract
FSUBR	Subtract real reversed
FSUBRP	Subtract real reversed and pop
FISUBR	Integer subtract reversed
Multiplication	
FMUL	Multiply real
FMULP	Multiply real and pop
FIMUL	Integer multiply
Division	
FDIV	Divide real
FDIVP	Divide real and pop
FIDIV	Integer divide
FDIVR	Divide real reversed
FDIVRP	Divide real reversed and pop
FIDIVR	Integer divide reversed
Other Operations	
FSQRT	Square root
FSCALE	Scale
FPREM	Partial reminder
FRNDINT	Round to integer
FXTRACT	Extract exponent and significand
FABS	Absolute value
FCHS	Change sign

Processor Control Instructions

FINIT/FNINIT	Initialize processor
FDISI/FNDISI	Disable interrupts
FENI/FNENI	Enable interrupts
FLDCW	Load control word
FSTCW/FNSTCW	Store control word
FSTSW/FNSTSW	Store status word
FCLEX/FNCLEX	Clear exceptions
FSTENV/FNSTENV	Store environment
FLDENV	Load environment
FSAVE/FNSAVE	Save state
FRSTOR	Restore state
FINCSTP	Increment stack pointer
FDECSTP	Decrement stack pointer
FFREE	Free register
FNOP	No operation
FWAIT	CPU wait

Table 3. Execution Time for Selected 8087 Actual and Emulated Instructions

Floating Point Instruction	Approximate Execution Time (microseconds)		
	8087 (5 MHz Clock)	8086 Emulation	8087 (8 MHz Clock)
Add/Subtract Magnitude	14/18	1,600	9/11
Multiply (single precision)	19	1,600	12
Multiply (extended precision)	27	2,100	17
Divide	39	3,200	24
Compare	9	1,300	6
Load (double precision)	10	1,700	6
Store (double precision)	21	1,200	13
Square Root	36	19,600	23
Tangent	90	13,000	56
Exponentiation	100	17,100	63

An NDP instruction either will not reference memory, will require loading one or more operands from memory into the NDP, or will require storing one or more operands from the NDP into memory. In the first case, a non-memory reference escape is used to start NDP operation. In the last two cases, the CU makes use of a "dummy read" cycle initiated by the CPU, in which the CPU calculates the operand address and initiates a bus cycle, but does not capture the data. Instead, the CPU captures and saves the address which the CPU places on the bus. If the instruction is a load, the CU additionally captures the data word when it becomes available on the local data bus. If data required is longer than one word, the CU immediately obtains the bus from the CPU using the request/grant protocol and reads the rest of the information in consecutive bus cycles. In a store operation, the CU captures and saves the store address as in a load, and ignores the data word that follows in the "dummy read" cycle. When the NDP is ready to perform the store, the CU obtains the bus from the CPU and writes the operand starting at the specified address.

Numeric Execution Unit

The NEU executes all instructions that involve the register stack. These include arithmetic, logical, transcendental, constant and data transfer instructions. The data path in the NEU is 80 bits wide (64 fraction bits, 15 exponent bits and a sign bit) which allows internal operand transfers to be performed at very high speeds.

When the NEU begins executing an instruction, it activates the NDP BUSY signal. This signal is used in conjunction with the CPU WAIT instruction to resynchronize both processors when the NEU has completed its current instruction.

Register Set

The NDP register set is shown in Figure 3. Each of the eight data registers in the NDP's register stack is 80 bits wide and is divided into "fields" corresponding to the NDP's temporary real data type. The register set may be addressed as a push down stack, through a top of stack pointer or any register may be addressed explicitly relative to the top of stack.

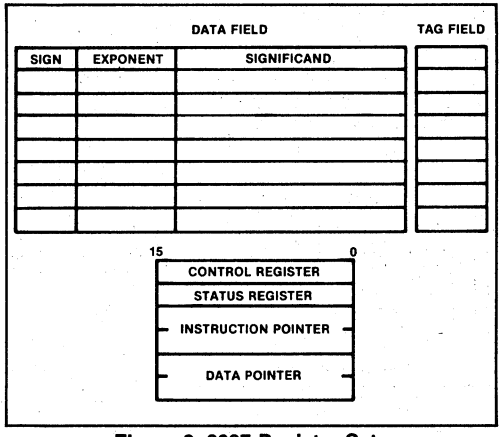


Figure 3. 8087 Register Set

Status Word

The status word shown in Figure 4 reflects the overall state of the NDP; it may be stored in memory and then inspected by CPU code. The status word is a 16-bit register divided into fields as shown in Figure 4. The busy bit (bit 15) indicates whether the NEU is executing an instruction (B = 1) or is idle (B = 0). Several

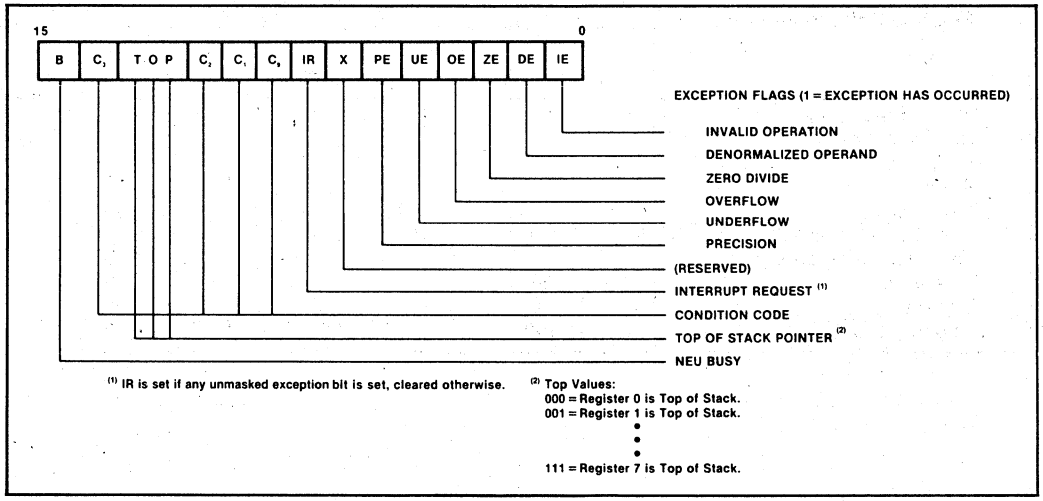


Figure 4. 8087 Status Word

instructions which store and manipulate the status word are executed exclusively by the CU, and these do not set the busy bit themselves.

The four numeric condition code bits (C_0-C_3) are similar to the flags in a CPU: various instructions update these bits to reflect the outcome of NDP operations.

Bits 13-11 of the status word point to the NDP register that is the current top-of-stack (TOP).

Bit 7 is the interrupt request bit. This bit is set if any unmasked exception bit is set and cleared otherwise.

Bits 5-0 are set to indicate that the NEU has detected an exception while executing an instruction.

Tag Word

The tag word marks the content of each register as shown in Figure 5. The principal function of the tag word is to optimize the NDP's performance. The tag word can be used, however, to interpret the contents of NDP registers.

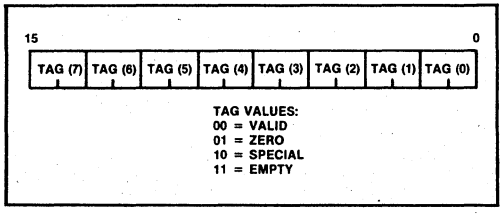


Figure 5. 8087 Tag Word

Instruction and Data Pointers

The instruction and data pointers (see Figure 6) are provided for user-written error handlers. Whenever the NDP executes an NEU instruction, the CU saves the instruction address, the operand address (if present) and the instruction opcode. The NDP can then store this data in memory.

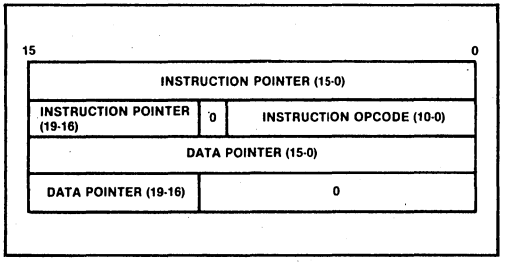


Figure 6. 8087 Instruction and Data Pointers

Control Word

The NDP provides several processing options which are selected by loading a word from memory into the control word. Figure 7 shows the format and encoding of the fields in the control word.

Exception Handling

The NDP detects six different exception conditions that can occur during instruction execution. Any or all exceptions will cause an interrupt if unmasked and interrupts are enabled.

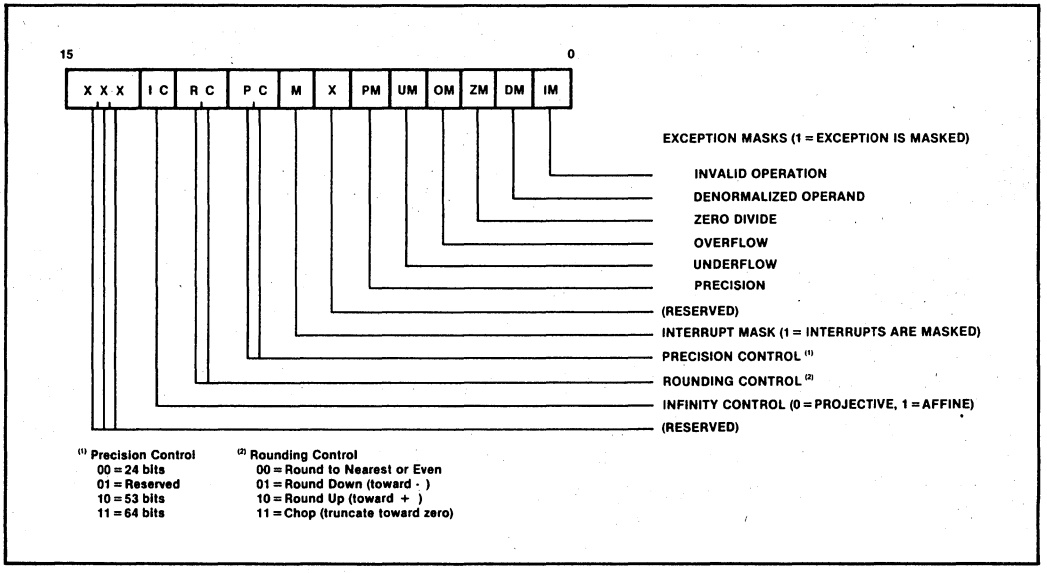


Figure 7. 8087 Control Word

If interrupts are disabled, the NDP will simply suspend execution until the host clears the exception. If a specific exception class is masked and that exception occurs however, the NDP will post the exception in the status register and perform an on-chip default exception handling procedure, thereby allowing processing to continue. The exceptions that the NDP detects are the following:

1. **INVALID OPERATION:** Stack overflow, stack underflow, indeterminate form (0/0, -, etc.) or the use of a Non-Number (NAN) as an operand. An exponent value is reserved and any bit pattern with this value in the exponent field is termed a Non-Number and causes this exception. If this exception is masked, the NDP default response is to generate a specific NAN called INDEFINITE, or to propagate already existing NANs as the calculation result.

2. **OVERFLOW:** The result is too large in magnitude to fit the specified format. The NDP will generate the code for infinity if this exception is masked.

3. **ZERO DIVISOR:** The divisor is zero while the dividend is a non-infinite, non-zero number. Again, the NDP will generate the code for infinity if this exception is masked.

4. **UNDERFLOW:** The result is non-zero but too small in magnitude to fit in the specified format. If this exception is masked the NDP will denormalize (shift right) the fraction until the exponent is in range. This process is called gradual underflow.

5. **DENORMALIZED OPERAND:** At least one of the operands or the result is denormalized; it has the smallest exponent but a non-zero significand. Normal processing continues if this exception is masked off.

6. **INEXACT RESULT:** If the true result is not exactly representable in the specified format, the result is rounded according to the rounding mode, and this flag is set. If this exception is masked, processing will simply continue.

SOFTWARE SUPPORT

The iSBC 337A/337 module is supported by the following Intel software products: iRMX™ 86 Operating System, iRMX 88 Real-time Multi-tasking Executive, ASM 86/88 Assembly language, PL/M 86/88 Systems Implementation Languages, Pascal 86/88, Fortran 86/88 along with iRMX Development Utilities Package. In addition to the instructions provided in the languages to support the additional math functions, a software emulator is also available to allow the execution of iAPX instructions without the need for the iSBC 337A/337 module. This allows for the development of software in an environment without the iAPX processor and then transporting to its final run time environment with no changes in software code or mathematical results.

SPECIFICATIONS

Physical Characteristics

Width — 5.33 cm (2.100")

Length — 5.08 cm (2.000")

Height — 1.82 cm (.718")
iSBC 337A board + host board

Weight — 17.33 grams (.576 oz.)

Electrical Characteristics

DC Power Requirements

$V_{CC} = 5V \pm 5\%$
$I_{CC} = 475$ mA max.
$I_{CC} = 350$ mA typ.

Environmental Characteristics

Operating Temperature — 0°C to 55°C with 200 linear feet/minute airflow

Relative Humidity — Up to 90% R.H. without condensation.

Reference Manual

147163-001 — iSBC 337A/337 MULTIMODULE Numeric Data Processor Hardware Reference Manual (NOT SUPPLIED WITH MULTIMODULE BOARD).

Manuals may be ordered from any Intel sales representative, distributor office, or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California, 95051.

ORDERING INFORMATION

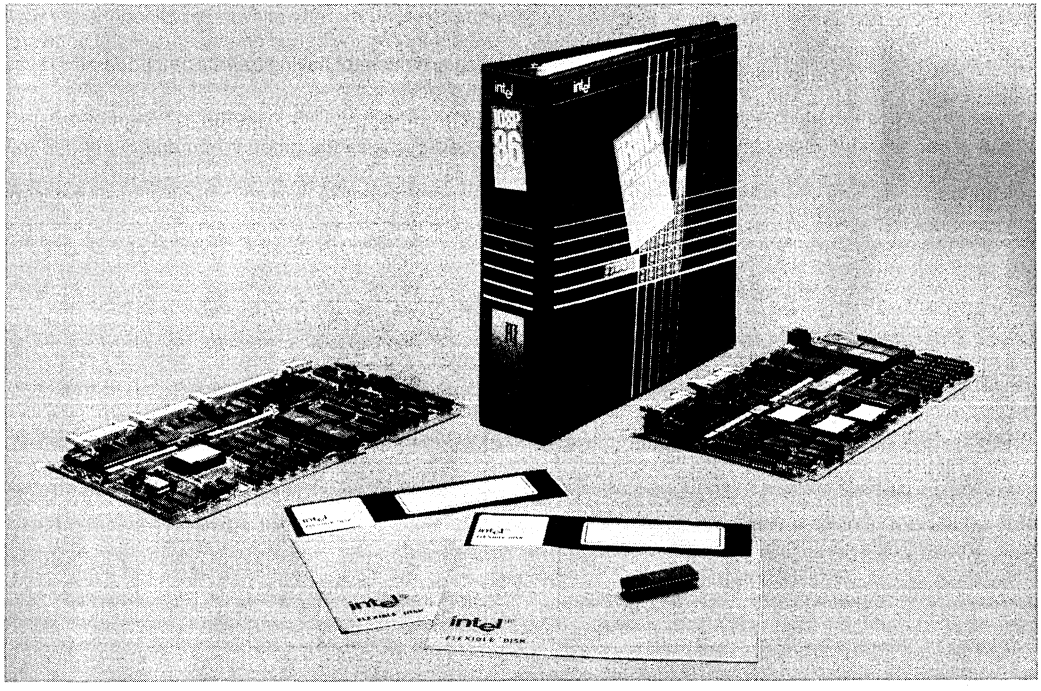
Part Number	Description
SBC 337A	MULTIMODULE Numeric Data Processor
SBC 337	MULTIMODULE Numeric Data Processor

System Software 6

iOSP™ 86
iAPX 86/30, iAPX 88/30, iAPX 186/30 and iAPX 188/30 SUPPORT PACKAGE

- **Development and run-time support for iAPX 86/30, 88/30, 186/30, and 188/30 Operating System Processors**
- **Total iRMX™ 86 Operating System software compatibility**
- **Extendable with iRMX™ 86 Operating System calls**
- **Compatible with Intel® PL/M 86, PASCAL 86, FORTRAN 86, and ASM86 MACRO ASSEMBLER**
- **Supports (P)ROM or RAM based system**
- **Supports custom system initialization**
- **Interactive Configuration Utility**

The Intel iOSP™ 86 Support Package for the iAPX 86/30, 88/30, 186/30, and 188/30 Operating System Processors contains a comprehensive set of easy-to-use tools needed to develop (P)ROM or RAM-based applications that use the 80130 Operating System Firmware component. This Support Package is compatible with all versions of the 80130 component. All of the system initialization and run-time facilities are provided in libraries that may be configured to specific requirements, and linked to application programs written in either ASM86 MACRO ASSEMBLER or a high level programming language such as PASCAL 86, FORTRAN 86, and PL/M 86. The iOSP 86 Package provides users with the basic initialization and interface routines needed to build application software based on the fundamental operating system functions of the iAPX 86/30, 88/30, 186/30, and 188/30 Operating System Processors. The iOSP 86 Package also enables users to add higher level I/O functions from the fully compatible iRMX™ 86 Operating System, or to form custom, real-time systems.



JUNE, 1984
ORDER NUMBER: 210236-002

FUNCTIONAL DESCRIPTION

The iAPX 86/30, 88/30, 186/30, and 188/30 Operating System Processors (OSPs) provide an easy-to-use foundation on which many real-time applications may be built. They provide the functions and system support needed to implement both simple and complex applications that require multiple tasks to run concurrently (see Figure 1). These services are made possible by the addition of the five new data types integrated into the 80130 Operating System Firmware (OSF) component. The 80130 OSF extends the basic data types of the CPU (integer, byte, character, etc.) by adding new system data types (JOB, TASK, MAILBOX, SEGMENT, and REGION), and extensive timer, interrupt, memory, and error management designed to give real-time response to multitasking and multiprogramming applications. As shown in the second half of the figure, other operating system functions such as mass storage I/O services and an easy-to-use Human Interface can be added easily, by using modules from the iRMX 86 Operating System. The iOSP 86 Support Package provides both an interface between application software and the Operating System Processors, and development tools designed to make the implementation and initialization of real-time, multitasking systems much easier.

The iOSP 86 Support package provides system developers with the configuration options necessary to tailor the iAPX 86/30, 88/30, 186/30, and 188/30 Operating System Processors to custom applications. Central to the entire configuration process is the Interactive Configuration Utility (ICU86). This utility is an easy-to-use tool which allows you to make configuration decisions by responding to screen-oriented displays. Using the ICU, users can build the necessary support code. The interface libraries form a simple interface between application software and the operating system primitives of the 80130 OSF component.

Memory and I/O Addressing

The 80130 OSF requires that a 16K byte block of memory address space be reserved for accessing internal functions. The ICU is used to specify the base address of the 80130 and the beginning of the initialization support code.

All Interrupt and Timer management of the OSF is controlled via a reserved 16 byte I/O address block that may be selected by the user. In addition, from 1 to 7 slave 8259A interrupt controllers can be specified in order to provide the system with up to 57 priority interrupt sources. The 80130 baud rate generator may also be configured to support an optional terminal interface.

Extending the 80130 OSF

The 80130 OSF allows users to add their own operating system extensions. These extensions may take advantage of the detailed and efficient intertask communication and synchronization primitives already provided by the 80130, and/or may utilize custom functions tailored to specific applications. The Support Package also enables users to extend the OSF with the extensive services of Intel's iRMX 86 Operating System, thereby allowing applications to grow without having to change or alter application software already written, or having to write other operating system software.

Use of the 80130 OSF with the iRMX 86 Operating System reduces the amount of memory needed for the iRMX 86 Nucleus layer by 14K bytes, and enables applications to take advantage of the increased

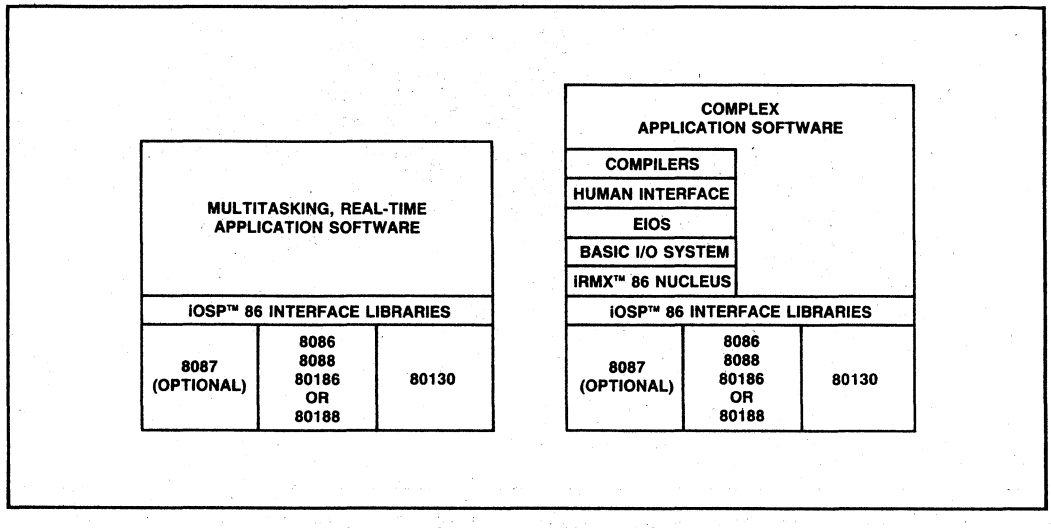


Figure 1. Structure of Typical Systems

performance and reduced size requirements inherent in the iAPX 86/30, 88/30, 186/30, and 188/30 Operating System Processors. Since each of the services provided by the 80130 component is totally compatible with iRMX 86, applications have an automatic upward path to support complete file systems and multiple processor environments.

Application Interfaces

Two interface libraries are included in the iOSP 86 Support Package. The first allows programmers to write application software modules in the Compact Model of computation supported by Intel's compilers. The second provides an interface to program segments written in either the Medium or Large Models. The iOSP 86 Support Package does not support program segments written in the Small Model.

The interface libraries provide the means of accessing all of the primitives supported by the Operating System Processors. With this interface, and all the memory management primitives of the OSPs, applications have full access to 1M byte of memory, and all of the addressing modes of the CPU.

These libraries are fully compatible with object modules produced by the ASM86 MACRO ASSEMBLER, and the PASCAL 86, FORTRAN 86, and PL/M 86 Compilers.

Application Initialization

The iOSP 86 Support Package provides, via the ICU, for the configuration of the system ROOT JOB, and all user application JOBs that require initialization when the system is started. The user also specifies the configuration of the interrupt system (including the optional iAPX 186/188 interrupt controller in either master or slave modes and any slave 8259A interrupt controllers) and the clock rate used for system timing. These choices are automatically programmed into the various devices when the system is initialized.

Parameter Validation

Parameter validation is a configuration option of an OSP-based system. The OSP can check the parameters of the primitive that you invoke either on a systemwide basis or on a per job basis.

Operating System Calls

The 80130 OSF performs a total of 38 operating system primitives all of which are completely compatible with the equivalent iRMX 86 Operating System calls. The iOSP 86 Support Package provides user-level interfaces to these primitives to enable applications to create, delete, control, and exchange the new data types provided by the 80130 OSF. In general, these interfaces allow application software to manage all of the resources of an iAPX 86/30, 88/30, 186/30, or 188/30 OSP (and an optional 8087 Numeric Processor Extension) system via any of the 38 system calls shown in Figure 2.

Required Development Hardware

Use of the iOSP 86 Support Package requires a Series III Intellec Development System with double density flexible diskette drives or any iRMX 86 system supporting a standard 5.25 inch or 8 inch flexible diskette drive and the iRMX 860 Assembler and Utilities Package. Use of the 80130 requires only a minimal system including either the iAPX 86/30, 88/30, 186/30 or 188/30 Operating System Processor, and enough system memory to contain the application programs and initialization and interface software provided in the iOSP 86 Package.

Board Level Product Support

Intel microcomputer boards which use the 80130 OSF include the iSBC 186/03 and the iSBC 186/51 Single Board Computers. An iOSP 86 application may be written specifically to run on these boards.

JOB GROUP
- CREATE JOB
- END INIT TASK

TASK GROUP
- CREATE TASK
- DELETE TASK
- SUSPEND TASK
- RESUME TASK
- SLEEP
- GET TASK TOKENS
- SET PRIORITY

MAILBOX GROUP
- CREATE MAILBOX
- DELETE MAILBOX
- SEND MESSAGE
- RECEIVE MESSAGE

SEGMENT GROUP
- CREATE SEGMENT
- DELETE SEGMENT

REGION GROUP
- CREATE REGION
- DELETE REGION
- SEND CONTROL
- RECEIVE CONTROL
- ACCEPT CONTROL

OBJECT MANAGEMENT GROUP
- CATALOG OBJECT
- LOOKUP OBJECT
- DISABLE DELETION
- ENABLE DELETION
- GET TYPE

INTERRUPT MANAGEMENT GROUP
- SET OS EXTENSION
- SET INTERRUPT
- ENTER INTERRUPT
- EXIT INTERRUPT
- WAIT INTERRUPT
- SIGNAL INTERRUPT
- RESET INTERRUPT
- ENABLE
- DISABLE
- GET LEVEL

ERROR CONTROL GROUP
- SET EXCEPTION
- SIGNAL EXCEPTION
- GET EXCEPTION

Figure 2. Operating System Primitives

Part Number	Description
OSP 86 B	iOSP 86 Support Package contained on an ISIS-II compatible, single-sided, double density 8 inch diskette.
OSP 86 E	iOSP 86 Support Package contained on an iRMX 86 format, single-sided, double density 8 inch diskette.
OSP 86 J	iOSP 86 Support Package contained on an iRMX 86 format double-sided double density, 5.25 inch, 48 tracks-per-inch diskette.

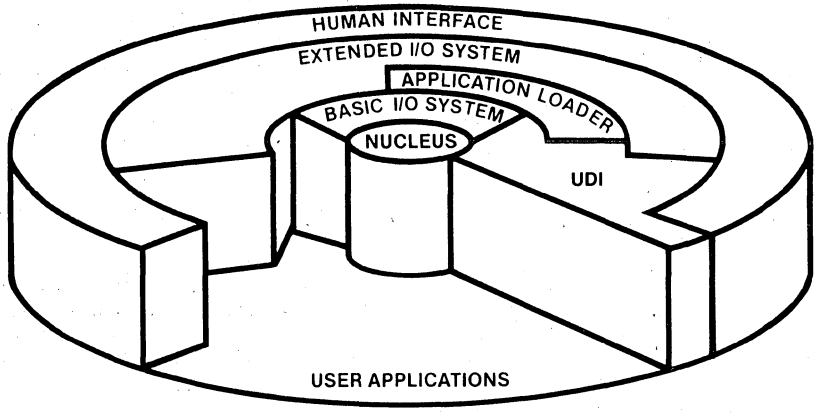
ORDERING INFORMATION

Each of the ordering options listed below include all the necessary initialization and interface procedures needed to use the iAPX 86/30, 88/30, 186/30, and 188/30 Operating System processors. Purchase of the iOSP 86 Package requires verification of an Intel Master Software License. Each package also includes an iOSP 86 User's Manual (Document Number 146798-001), and a 90 day update service.

iRMX™ 86 OPERATING SYSTEM

- **Real-time processor management for time-critical iAPX 86, iAPX 88, iAPX 186, iAPX 188, and iAPX 286 (Real Address Mode) applications**
- **On-target system development with Universal Development Interface (UDI)**
- **Configurable system size and function for diverse application requirements**
- **All iRMX™ 86 code can be (P)ROM'ed to support totally solid state designs**
- **Compatible operating system services for iAPX 86/30, 88/30, 186/30 and 188/30 Operating System Processors (iOSP™ 86)**
- **Configured systems for the iAPX 86 and iAPX 286 processors in Intel integrated system products (iSYS 86/300 and iSYS 286/300)**
- **Multi-terminal support with multi-user human interface**
- **Broad range of device drivers included for industry standard MULTIBUS® peripheral controllers**
- **Complete support of 8087 and 80287 processor extension**
- **Powerful utilities for interactive configuration and real-time debugging**

The iRMX™ 86 Operating System is an easy-to-use, real-time, multi-tasking and multi-programming software system designed to manage and extend the resources of iSBC® 86, iSBC 88, iSBC 186, iSBC 188, and iSBC 286 Single Board Computers, as well as other iAPX 86, iAPX 88, iAPX 186, iAPX 188, and iAPX 286 (Real Address Mode) based microcomputers. iRMX 86 functions are available in silicon with the iAPX 86/30, 88/30, 186/30 and 188/30 Operating System Processors, in a user configurable software package. iRMX 86 functions are also fully integrated into the SYSTEM 86/300 and SYSTEM 286/300 Family of Microcomputer Systems. The Operating System provides a number of standard interfaces that allow iRMX 86 applications to take advantage of industry standard device controllers, hardware components, and a number of software packages developed by Independent Software Vendors (ISVs). Many high-performance features extend the utility of iRMX 86 Systems into applications such as data collection, transaction processing, and process control where immediate access to advances in VLSI technology is paramount. These systems may deliver real-time performance and explicit control over resources; yet also support applications with multiple users needing to simultaneously access terminals. The configurable layers of the System provide services ranging from interrupt management and standard device drivers for many sophisticated controllers, to data file maintenance commands provided by a comprehensive multi-user human interface. By providing access to the standard Universal Development Interface (UDI) for each user terminal, Original Equipment Manufacturers (OEMs) can pass program development and target application customization capabilities to their users.



iRMX™ VLSI Operating System

April, 1984
Order Number: 210885-002

The iRMX 86 Operating System is a complete set of system software modules that provide the resource management functions needed by computer systems. These management functions allow Original Equipment Manufacturers (OEMs) to best use resources available in microcomputer systems while getting their products to market quickly, saving time and money. Engineers are relieved of writing complex system software and can concentrate instead on their application software.

This data sheet describes the major features of the iRMX 86 Operating System. The benefits provided to engineers who write application software and to users who want to take advantage of improving microcomputer price and performance are explained. The first section outlines the system resource management functions of the Operating System and describes several system calls. The second section gives a detailed overview of iRMX 86 features aimed at serving both the iRMX 86 system designer and programmer, as well as the end users of the product into which the Operating System is incorporated.

FUNCTIONAL DESCRIPTION

To take best advantage of iAPX 86, 88, 186, 188, and 286 (Real Address Mode) microprocessors in applications where the computer is required to perform many functions simultaneously, the iRMX 86 Operating System provides a multiprogramming environment in which many independent, multi-tasking application programs may run. The flexibility of independent environments allows application programmers to separately manage each application's resources during both the development and test phases.

The resource management functions of the iRMX 86 System are supported by a number of configurable software layers. While many of the functions supplied by the innermost layer, the Nucleus, are required by all systems, all other functions are optional. The I/O systems, for example, may be omitted in systems having no secondary storage requirement. Each layer provides functions that encourage application programmers to use modular design techniques for quick development of easily maintainable programs.

The components of the iRMX 86 Operating System provide both implicit and explicit management of system resources. These resources include processor scheduling, up to one megabyte of system memory, up to 57 independent interrupt sources, all input and output devices, as well as directory and data files contained on mass storage devices and accessed by a number of independent users. Management of these system resources and methods for sharing resources between multiple processors and users is discussed in the following sections.

Process Management

To implement multi-tasking application systems, programmers require a method of managing the different processes of their application, and for allowing the processes to communicate with each other. The Nucleus layer of the iRMX 86 System provides a number of facilities to efficiently manage these processes, and to effectively communicate between them. These facilities are provided by system calls that manipulate data structures called tasks, jobs, regions, semaphores and mailboxes. The iRMX 86 System refers to these structures as "objects".

Tasks are the basic element of all applications built on the iRMX 86 Operating System. Each task is an entity capable of executing CPU instructions and issuing system calls in order to perform a function. Tasks are characterized by their register values (including those of an optional 8087 or 80287 Numeric Processor Extension), a priority between 0 and 255, and the resources associated with them.

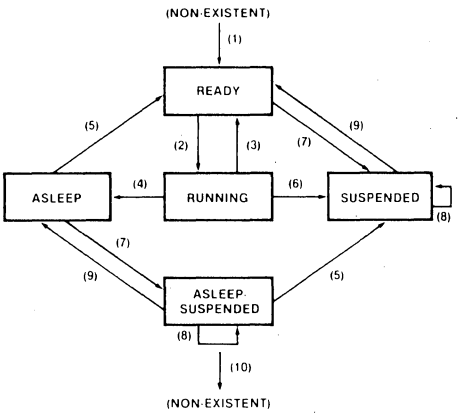
Each iRMX 86 task in the system is scheduled for operation by the iRMX 86 Nucleus. Figure 1 shows the five states in which each task may be placed, and some examples of how a task may move from one state to another. The iRMX 86 Nucleus ensures that each task is placed in the correct state, defined by the events in its external environment and by the task issuing system calls. Each task has a priority to indicate its relative importance and need to respond to its environment. The Nucleus guarantees that the highest priority ready-to-run task is the task that runs.

Jobs are used to define the operating environment of a group of tasks. Jobs effectively limit the scope of an application by collecting all of its tasks and other objects into one group. Because the environment for execution of an application is defined by an iRMX 86 job, separate applications can be efficiently developed by separate development teams.

The iRMX 86 Operating System provides two primary techniques for real-time event synchronization in multitask applications: regions and semaphores.

Regions are used to restrict access to critical sections of code and data. Once the iRMX 86 Operating System gives a task access to resources guarded by a region, no other tasks may make use of the resources, and the task is given protection against deletion and suspension. Regions are typically used to protect data structures from being simultaneously updated by multiple tasks.

Semaphores are used to provide mutual exclusion between tasks. They contain abstract "units" that are sent between the tasks, and can be used to implement the cooperative sharing of resources.



NOTES:
(1) Task is created
(2) Task becomes highest priority ready task
(3) Task gets pre-empted by one with higher priority
(4) Task calls SLEEP or task waits at an exchange
(5) Task sleep period has ended, message was sent to waiting task or wait has ended
(6) Task calls SUSPEND on self
(7) Task suspended by other than self
(8) Task suspended by other than self or a resume that did not bring suspension depth to zero
(9) Task was resumed by other task
(10) Task is deleted

Figure 1. Task State Diagram

Multi-tasking applications must communicate information and share system resources among cooperating tasks. The iRMX 86 Operating System assigns a unique 16-bit number, called a token, to each object created in the System. Any task in possession of this token is able to access the object. The iRMX 86 Nucleus allows tasks to gain access to objects, and hence system resources, at run-time with two additional mechanisms: mailboxes and object directories.

Mailboxes are used by tasks wishing to share objects with other tasks. A task may share an object by sending the object token via a mailbox. The receiving task can check to see if a token is there, or can wait at the mailbox until a token is present.

Object Directories are also used to make an object available to other tasks. An object is made public by cataloging its token and name in a directory. In this manner, any task can gain access to the object by knowing its name, and job environment that contains the directory.

Two example jobs are shown in Figure 2 to demonstrate how two tasks can share an object that was not

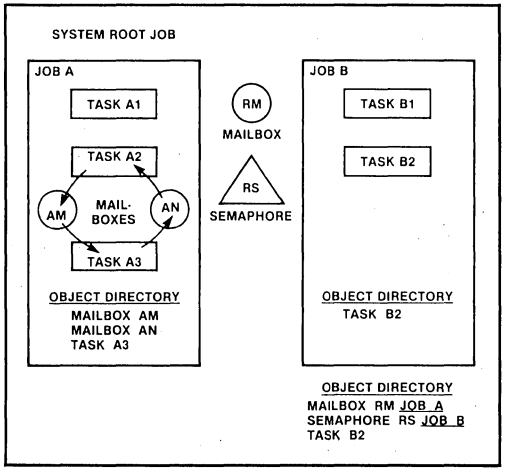


Figure 2. Multiple Jobs Example

known to the programmer at the time the tasks were developed. Both Job 'A' and Job 'B' exist within the environment of the 'Root Job' that forms the foundation of all iRMX 86 systems. Each job possesses a directory in which tasks may catalog the name of an object. Semaphore 'RS', for example, is accessable by all tasks in the system, because its name is cataloged in the directory of the Root Job. Mailbox 'AN' can be used to transfer objects between Tasks 'A2' and 'A3' because its token is accessable in the object directory for Job 'A'.

Table 1 lists the major functions of the iRMX 86 Nucleus that manage system processes.

Memory Management

Each job in an iRMX 86 System defines the amount of the one megabyte of addressable memory to be used by its tasks. The iRMX 86 Operating System manages system memory and allows jobs to share this critical resource by providing another object type: segments.

Segments are contiguous pieces of memory between 16 Bytes and 64K Bytes in length, that exist within the environment of the job in which they were created. Segments form the fundamental piece of system memory used for task stacks, data storage, system buffers, loading programs from secondary storage, passing information between tasks, etc.

The example in Figure 2 also demonstrates when information is shared between Tasks 'A2' and 'A3'; 'A2' only needs to create a segment, put the information in the memory allocated, and send it via the Mailbox 'AM' using the RQ$SEND$MESSAGE system call (see Table 1). Task 'A3' would get the message by using the RQ$RECEIVE$MESSAGE system call. The Figure also shows how the receiving task could signal the sending task by sending an acknowledgement via the second Mailbox 'AN'.

Table 1. Process Management System Calls

System Call	Function Performed
RQ$CREATE$JOB	Creates an environment for a number of tasks and other objects, as well as creating an initial task and its stack.
RQ$DELETE$JOB	Deletes a job and all the objects currently defined within its bounds. All memory used is returned to the job from which the deleted job was created.
RQ$OFFSPRING	Provides a list of all the current jobs created by the specified job.
RQ$CATALOG$OBJECT	Enters a name and token for an object into the object directory of a job.
RQ$UNCATALOG$OBJECT	Removes an object's token and its name from a job's object directory.
RQ$LOOKUP$OBJECT	Returns a token for the object with the specified name found in the object directory of the specified job.
RQGETTYPE	Returns a code for the type of object referred to by the specified token.
RQ$CREATE$MAILBOX	Creates a mailbox with queues for waiting tasks and objects with FIFO or PRIORITY discipline.
RQ$DELETE$MAILBOX	Deletes a mailbox.
RQ$SEND$MESSAGE	Sends an object to a specified mailbox. If a task is waiting, the object is passed to the appropriate task according to the queuing discipline. If no task is waiting, the object is queued at the mailbox.
RQ$RECEIVE$MESSAGE	Attempts to receive an object token from a specified mailbox. The calling task may choose to wait for a specified number of system time units if no token is available.
RQ$DISABLE$DELETION	Prevents the deletion of a specified object by increasing its disable count by one.
RQ$ENABLE$DELETION	Reduces the disable count of an object by one, and if zero, enables deletion of that object.
RQ$FORCE$DELETE	Forces the deletion of a specified object if the disable count is either 0 or 1.
RQ$CREATE$TASK	Creates a task with the specified priority and stack area.
RQ$DELETE$TASK	Deletes a task from the system, and removes it from any queues in which it may be waiting.
RQ$SUSPEND$TASK	Suspends the operation of a task. If the task is already suspended, its suspension depth is increased by one.
RQ$RESUME$TASK	Resumes a task. If the task had been suspended multiple times, the suspension depth is reduced by one, and it remains suspended.
RQ$SLEEP	Causes a task to enter the ASLEEP state for a specified number of system time units.
RQGETTASK$TOKENS	Gets the token for the calling task or associated objects within its environment.
RQSETPRIORITY	Dynamically alters the priority of the specified task.
RQGETPRIORITY	Obtains the current priority of a specified task.
RQ$CREATE$REGION	Creates a region, with an associated queue of FIFO or PRIORITY ordering discipline.
RQ$DELETE$REGION	Deletes the specified region if it is not currently in use.
RQ$ACCEPT$CONTROL	Gains control of a region only if the region is immediately available.
RQ$RECEIVE$CONTROL	Gains control of a region. The calling task may specify the number of system time units it wishes to wait if the region is not immediately available.
RQ$SEND$CONTROL	Relinquishes control of a region.
RQ$CREATE$SEMAPHORE	Creates a semaphore.
RQ$DELETE$SEMAPHORE	Deletes a semaphore.
RQ$SEND$UNITS	Increases a semaphore counter by the specified number of units.
RQ$RECEIVE$UNITS	Attempts to gain a specified number of units from a semaphore. If the units are not immediately available, the calling task may choose to wait.

Each job is created with both maximum and minimum limits set for its memory pool. Memory required by all objects and resources created in the job is taken from this pool. If more memory is required, a job may be allowed to borrow memory from the pool of its containing job (the job from which it was created). In this manner, initial jobs may efficiently allocate memory to jobs they subsequently create, without knowing their exact requirements.

The iRMX 86 Operating System supplies other memory managment functions to search specific address ranges for available memory. The System performs this search at system initialization, and can be configured to ignore non-existent memory and addresses reserved for I/O devices and other application requirements.

Table 2 lists the major system calls used to manage the system memory.

Interrupt Management

Real-time systems, by their nature, must respond to asynchronous and unpredictable events quickly. The iRMX 86 Operating System uses interrupts and the event-driven Nucleus described earlier to give real-time response to events. Use of a pre-emptive scheduling technique ensures that the servicing of high priority events always takes precedence over other system activities.

The iRMX 86 Operating System gives applications the flexibility to optimize either interrupt response time or interrupt response capability by providing two tiers of Interrupt Management. These two distinct tiers are managed by Interrupt Handlers and Interrupt Tasks.

Interrupt Handlers are the first tier of interrupt service. For small simple functions, interrupt handlers are often the most efficient means of responding to an event. They provide faster response than interrupt tasks, but must be kept simple since interrupts (except the iAPX 86, 88, 186, 188, and 286 non-maskable interrupt) are masked during their execution. When extended service is required, interrupt handlers "signal" a waiting interrupt task that, in turn, performs more complicated functions.

Interrupt Tasks are distinct tasks whose priority is associated with a hardware interrupt level. They are permitted to make any iRMX 86 system call. While an interrupt task is servicing an interrupt, interrupts of lower priority are not allowed to pre-empt the system.

Table 3 shows the iRMX 86 System Calls provided to manage interrupts.

Table 2. Memory Management System Calls

System Call	Function Performed
RQ$CREATE$SEGMENT	Dynamically allocates a memory segment of the specified size.
RQ$DELETE$SEGMENT	Deletes the specified segment by deallocating the memory.
RQGETPOOL$ATTRIBUTES	Returns attributes such as the minimum and maximum, as well as current size of the memory in the environment of the calling task's job.
RQGETSIZE	Returns the size (in bytes) of a segment.
RQSETPOOL$MIN	Dynamically changes the minimum memory requirements of the job environment containing the calling task.

Table 3. Interrupt Management System Calls

System Call	Function Performed
RQSETINTERRUPT	Assigns an interrupt handler and, if desired, an interrupt task to the specified interrupt level. Usually the calling task becomes the interrupt task.
RQ$RESET$INTERRUPT	Disables an interrupt level, and cancels the assignment of the interrupt handler for that level. If an interrupt task was assigned, it is deleted.
RQGETLEVEL	Returns the number of the highest priority interrupt level currently being processed.
RQ$SIGNAL$INTERRUPT	Used by an interrupt handler to signal the associated interrupt task that an interrupt has occurred.
RQ$WAIT$INTERRUPT	Used by an interrupt task to SLEEP until the associated interrupt handler signals the occurrence of an interrupt.
RQ$EXIT$INTERRUPT	Used by an interrupt handler to relinquish control of the System.
RQ$ENABLE	Enables the hardware to accept interrupts from a specified level.
RQ$DISABLE	Disables the hardware from accepting interrupts at or below a specified level.

INTERRUPT MANAGEMENT EXAMPLE

Figure 3 illustrates how the iRMX 86 Interrupt System may be used to output strings of characters to a printer. In the example, a mailbox named 'PRINT' is used by all tasks in the system to queue messages to be printed. Application tasks put the characters in segments that are transmitted to the printer interrupt task via the PRINT Mailbox. Once printing is complete, the same interrupt task passes the messages on to another application via the FINISHED Mailbox so that an operator message can be displayed.

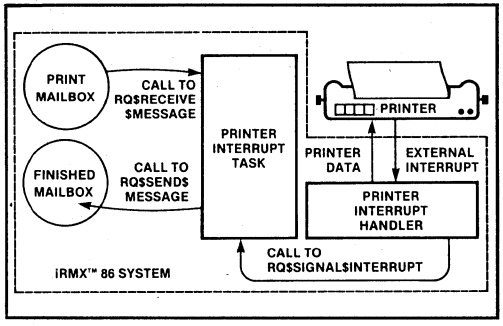


Figure 3. Interrupt Management Example

Basic I/O System

The Basic I/O System (BIOS) provides the direct access to I/O devices needed by real-time applications. The BIOS allows I/O functions to overlap other system functions. In this manner, application tasks make asynchronous calls to the iRMX 86 BIOS, and proceed to perform other activities. When the I/O request must be completed before an application can continue, the task waits at a mailbox for the result of the operation.

Some system calls provided by the BIOS are listed in Table 4.

The Basic I/O System communicates with peripheral devices through device drivers. These device drivers provide the System with four basic functions needed to control and communicate with devices: Initialize I/O, Finish I/O, Queue I/O, and Cancel I/O. Using the device driver interface, users of non-standard devices may write custom drivers compatible with the I/O System.

The iRMX 86 Operating System includes a number of device drivers to allow applications to use standard USART serial communications devices, multiple CRTs and keyboards, bubble memories, diskettes, disks, a Centronics-type parallel printer, and many of Intel's iSBC and iSBX™ device controllers (see Table 8). If an application requires use of a non-standard device, users need only write a device driver to be included with the BIOS, and access it as if it were part of the standard system. For most common random-access devices, this job is further simplified by using standard routines provided with the System. Use of this technique ensures that applications can remain device independent.

Multi-Terminal Support

The iRMX 86 Terminal Support provides line editing and terminal control capabilities. The Terminal Support communicates with devices through simple drivers that do only character I/O functions. Dynamic terminal reconfiguration is provided so that attributes such as terminal type and line speed may be changed without modifying the application or the Operating System. Dynamic configuration may be typed in, generated programmatically or stored in a file and copied to a terminal I/O connection.

Table 4. Key BIOS I/O Management System Calls

System Call	Function Performed
RQAATTACH$FILE	Creates a Connection to an existing file.
RQACHANGE$ACCESS	Changes the types of accesses permitted to the specified user(s) for a specific file.
RQACLOSE	Closes the Connection to the specified file so that it may be used again, or so that the type of access may be changed.
RQACREATE$DIRECTORY	Creates a Named File used to store the names and locations of other Named Files.
RQACREATE$FILE	Creates a data file with the specified access rights.
RQADELETE$CONNECTION	Deletes the Connection to the specified file.
RQAGET$FILE$STATUS	Returns the current status of a specified file.
RQAOPEN	Opens a file for either read, write, or update access.
RQAREAD	Reads a number of bytes from the current position in a specified file.
RQASEEK	Moves the current data pointer of a Named or Physical file.
RQAWRITE	Writes a number of bytes at the current position in a file.
RQ$WAIT$IO	Synchronizes a task with the I/O System by causing it to wait for I/O operation results.

The iRMX 86 Terminal Support provides automatic translation of control characters to specific control sequences for each terminal. This translation enables applications using standard control characters to function with non-standard terminals. The translation requirements for each terminal can be stored in terminal description files and copied to a connection, as described above.

Disk I/O Performance

Figure 4 shows iRMX 86 performance obtained using the iSBC 215 Winchester Disk and iSBX 218A Diskette Controllers under the specified conditions. The vertical axis is a linear scale of throughput in units of 10,000 bytes per second. The horizontal axis is a logrithmic scale showing the transfer size for the reads and writes. Each data point on the graph indicates the time required for a read/write request of 64K bytes. Therefore each transfer size on the horizontal scale less than 64K was repeated until a total request of 64K was read or written.

Each device driver can be used to interface to a number of separate and, in some cases, different devices (see Figure 5). The iSBC 215 Device Driver, supplied with the system, is capable of supporting the iSBC 215 Winchester Disk Controller, the iSBC 220 SMD Disk Controller, and the iSBX 218A Flexible Disk Controller (when mounted on an iSBC 215 board). Each device controller may, in turn, control a number of separate device units. In addition, each driver may control a number of like device controllers. This capability allows the use of large storage systems with a minimum of I/O system code to write or maintain.

Extended I/O System

The iRMX 86 Extended I/O System (EIOS) adds a number of I/O management capabilities to simplify access to files. Whereas the BIOS provides users with the basic system calls needed for direct management of I/O resources, many users prefer to have the system perform all the buffering and synchronization of I/O requests automatically. The EIOS allows users to access I/O devices without having to write procedures for buffering data, or to specify particular devices with constant device names.

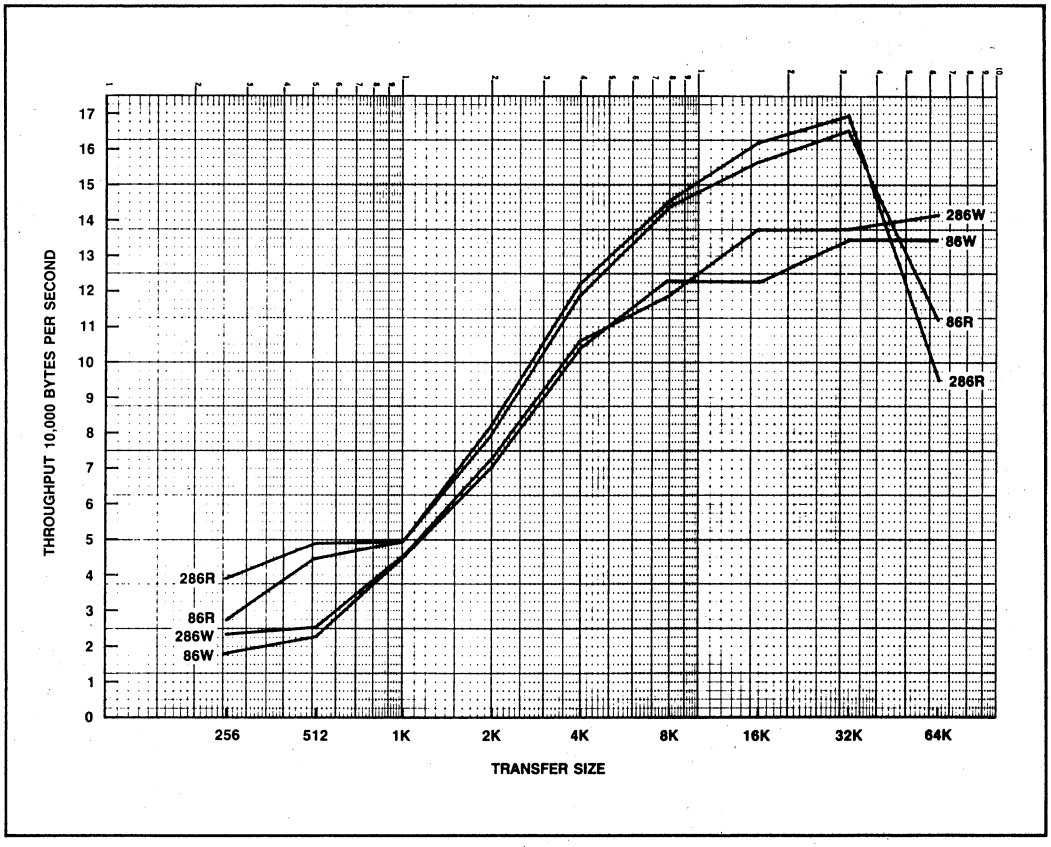


Figure 4. iRMX™ 86 Disk I/O Performance

Order Number: 210885-002

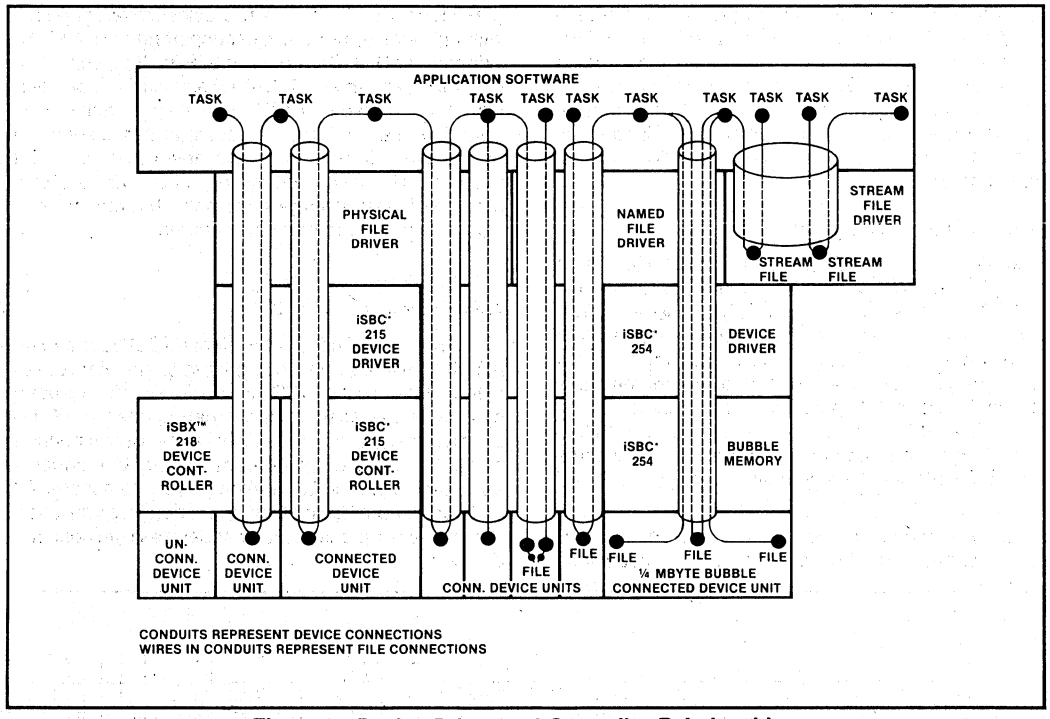


Figure 5. Device Driver and Controller Relationships

By performing device buffering automatically, the iRMX 86 EIOS optimizes accesses to disks and other devices. Often, when an application task asks the System to READ a portion of a file, the System is able to respond immediately with the data it has read in advance of the request. Similarly, the EIOS will not delay a task for writing data to a device unless it is specifically told to, or if its output buffers are filled.

Logical file and device names are provided by the EIOS to give applications complete file and device independence. Applications may send data to the 'line printer' (:LP:) without needing to know which specific device will be used as the printer. This logical name may, in fact, not be a printer at all, but it could be a disk file that is later scheduled for printing.

The EIOS uses the functions provided by the BIOS to synchronize individual I/O requests with results returned by device drivers. Most EIOS system calls are similar to the BIOS calls, except that they appear to suspend the operation of the calling task until the I/O requests are completed.

Two new primitives have been added to the EIOS. These are: RQ$HYBRID$DETACH$DEVICE and RQ$GET$LOGICAL$DEVICE$STATUS.

RQ$HYBRID$DETACH$DEVICE allows a programmer to temporarily detach a device physically so it can be temporarily attached another way.

RQGETLOGICAL$DEVICE$STATUS provides information about a logical device: the physical device name, file driver, number of connections to the device, and the owner of the device.

File Management

The iRMX 86 Operating System provides three distinct types of files to ensure efficient management of both program and data files: Named Files, Physical Files, and Stream Files. Each file type provides access to I/O devices through the standard device drivers mentioned earlier. The same device driver is used to access physical and named files for a given device.

NAMED FILES

Named files allow users to access information on secondary storage by referring to a file with its ASCII name. The names of files stored on a device are stored in special files called directories. As directories are themselves named files, the iRMX 86 File System allows directories to contain the names of other directories. Figure 6 illustrates the resulting hierarchical file structure. This structure is useful for isolating file names to particular user applications, and for tailoring system data to the requirements of users and applications sharing storage devices. Using different branches on the directory tree, different users do not have to coordinate in naming their files to ensure unique names.

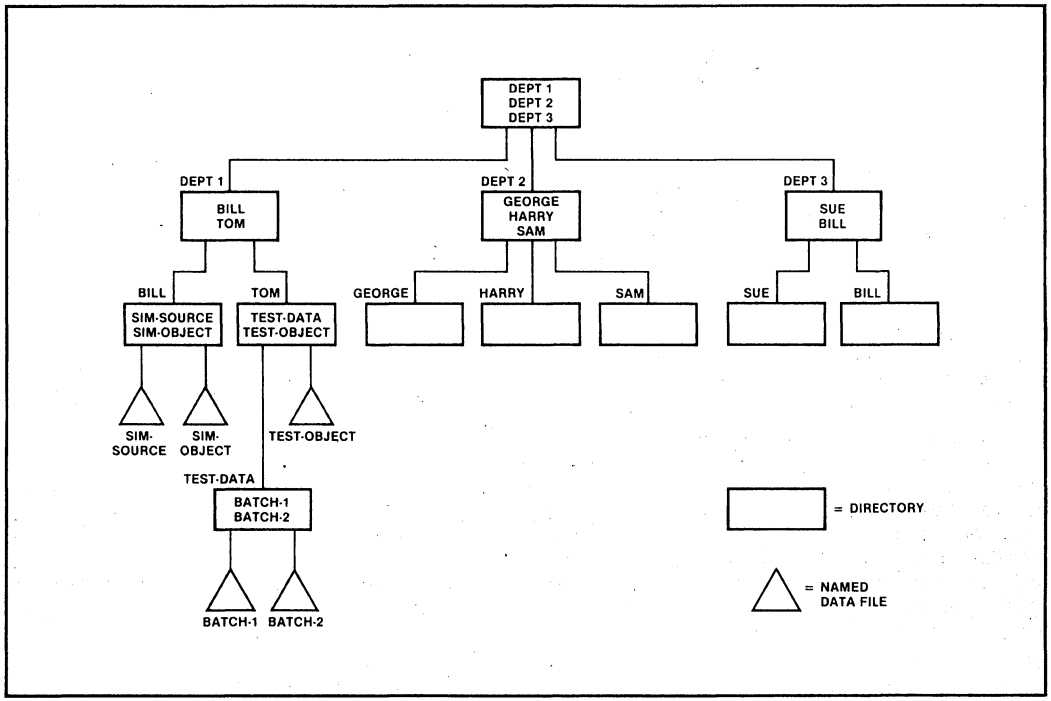


Figure 6. Hierarchical Named File Structure

Whenever a request is made involving a file name, the System will search the appropriate directory in order to find the necessary information about the file's size, access rights, and specific location on the storage device.

The iRMX 86 BIOS uses an efficient format for writing the directory and data information into secondary storage. This standard iRMX 86 format is fully compatible with the ISO Media standard, and other Intel systems such as the iRMX 88 Operating System. This structure enables the system to directly access any byte in a file, often without having to do additional I/O to access space allocation information. The maximum size of an individual file is 4.3 billion bytes.

EASE OF ACCESS

The hierarchical file structure is provided to isolate and organize collections of named files. To give operators fast and simple access to any level within the file tree, an ATTACHFILE command is provided. This command allows operators to create a logical name to a point in the tree so that a long sequence of characters need not be typed each time a file is referred to.

ACCESS PROTECTION

Access to each Named File is protected by the rights assigned to each user by the owner of the file. Rights to read, append, update, and delete may be selectively granted to other users of the system. In general, users of Named Files are classified into one of two categories: User and World. Users are used when different programmers and programs need to share information stored in a file. The World classification is used when rights are to be granted to all who can use the system.

PHYSICAL FILES

Physical Files allow more direct device access than Named Files. Each Physical File occupies an entire device, treated as a single stream of individually accessable bytes. No access control is provided for Physical Files as they are typically used for such applications as driving a printing device, translating from one device format to another, driving a paper tape device, real-time data acquisition, and controlling analog mechanisms.

STREAM FILES

Stream Files provide applications with a method of using iRMX 86 file management methods for data that does not need to go into secondary storage. Stream Files act as direct channels, through system memory, from one task to another. These channels are very useful to programs, for example, wishing to preserve file and device independence allowing data sent to a printer one time, to a disk file another time, and to another program on a different occasion.

Order Number: 210885-002

BOOTSTRAP AND APPLICATION LOADERS

Two utilities are supplied with the System to load programs and data into system memory from secondary storage devices:

The iRMX 86 Bootstrap Loader can be configured to a size of less than 1K Bytes of P(ROM), and is typically used to load the initial system from the system disk into memory, and begin its execution. Error reporting and debug switch features have been added to the Bootstrap Loader. When the Bootstrap Loader detects errors such as: file does not exist or device not ready, an error message is reported back to the user. The debug switch will cause the Bootstrap Loader to load the system but not begin its execution. Instead the Bootstrap Loader will pass control to the monitor at the first instruction to be executed by the system.

The Application Loader is typically used by application programs already running in the system to load additional programs and data from any secondary storage device. The Human Interface layer, for example, uses the Application Loader to load the non-resident Human Interface Commands. The Application Loader is capable of loading both relocatable and absolute code as well as program overlays.

Human Interface

The flexibility of the interface between computer controlled machines and their users often determines the usability and ultimate success of the machines. Table 11 lists iRMX 86 Human Interface functions giving users and applications simple access to the file and system management capabilities described earlier. The process, interrupt, and memory managment functions described earlier, are performed automatically for Human Interface users.

MULTI-USER ACCESS

Using the multi-terminal support provided by the BIOS, the iRMX 86 Human Interface can support several simultaneous users. The real-time nature of the system is maintained by providing a priority for each user, and using the event-driven iRMX 86 Nucleus to schedule tasks. High-performance interrupt response is guaranteed even while users interact with various application packages. For example, multi-terminal support allows one person to be using the iRMX 86 Editor, while another compiles a FORTRAN 86 or PASCAL 86 program, while several others load and access applications.

Each terminal attached to the iRMX 86 multi-user Human Interface is automatically associated with a user, a memory pool, and an initial program to run when the terminal is connected. This association is made using a file that may be changed at any time. Changes are effective the next time the system is initialized.

The initial program specified for each terminal can be a special application program, a custom Human Interface, or the standard iRMX 86 Command Line Interpreter (CLI). For example, you may choose to use the Microsoft Basic Interpreter as this initial program. After system start-up, each terminal user would be able to run the interpreter without asking for it to be loaded. From the BASIC interpreter, an operator, for example, could run a data collection program, written in BASIC, that communicates with several laboratory instruments, and prints charts and reports based on certain test results. When finished entering, changing, or running a BASIC program, the terminal would remain in BASIC for the next user.

Specifying an application program as a terminal's initial program makes the interface between operators and the computer system much simpler. Each operator need only be aware of the function of a particular application; not needing to interact with any unfamiliar functions also available on the application system.

Specifying the standard iRMX 86 Human Interface CLI as the initial program enables users of the terminals to access all iRMX 86 functions. This CLI makes it easy to manage iRMX 86 files, load and execute Intel-supplied and custom programs, and submit command files for execution.

FEATURE OVERVIEW

The iRMX 86 Operating System is well suited to serve the demanding needs of real-time applications executing on complex microprocessor systems. The iRMX 86 System also provides many tools and features needed by real-time system developers and programmers. The following sections describe features useful in both the development and execution environments. The description of each feature outlines the advantages given to hardware and software engineers concerned with overall system cost, expandability with custom and industry standard options, and long-term maintenance of iRMX 86-based systems. The development environment features also describe the ease with which the iRMX 86 Operating System can be incorporated into overall system designs.

Execution Environment Features

REAL-TIME PERFORMANCE

The iRMX 86 Operating System is designed to offer the high performance, multi-tasking functions required by real-time systems. Designers can make use of the latest VLSI devices such as the 8087 or 80287 Numeric Processor Extension, and the 80130 Operating System Firmware Component to improve their system cost/performance ratio or the iMMX™ 800 MULTIBUS® Message Exchange software package to divide and coordinate various system activities among multiple processors. Typical iRMX 86 system performance characteristics are shown in Table 5.

Many real-time systems require high performance operation. To meet this requirement, all of iRMX 86 can be put into zero wait-state P(ROM). This approach eliminates the possibility of disk access times slowing down performance, while allowing system designers to take advantage of high performance memory devices.

CONFIGURABILITY

The iRMX 86 Operating System is configurable by system layer, and by system call within each layer. In addition all the I/O port addresses used by the System are configurable by the user. This flexibility gives designers the freedom to choose configurations of hardware and software that best suit their size and functional requirements. Two example configurations are shown in Figure 7.

Table 5. iRMX™ Real-Time Performance Using iSBC® 86/30 and iSBC® 286/10 Single Board Computers

Real-Time Function	iSBC® 86/30 Execution Time (msec)	iSBC® 286/10 Execution Time (msec)
Suspend Task	1.02	0.83
Interrupt Latency (to handler)	0.29 (Max)	0.20 (Max)
Interrupt Latency (to handler)	0.02 (Typical)	0.03 (Typical)
Context Switch Caused By Interrupt	0.84 (Max)	0.78 (Max)
Send Message (no context switch)	0.32	0.25
Send Message (with context switch)	0.58	0.49
Send Control (no context switch)	0.21	0.16
Send Control (with context switch)	0.64	0.54
Receive Control (no waiting)	0.26	0.19

Context switch time is the time between executing in the context of a task, and the first instruction to execute in the context of another task.

The execution times shown in Column 2 were measured using an 8MHz iSBC Single Board Computer, 256K on-board RAM, and all program and data stored in on-board RAM.

The execution times shown in Column 3 were measured using a 5MHz iSBC 286/10 Single Board Computer, no on-board RAM, and all program and data stored in LBX RAM.

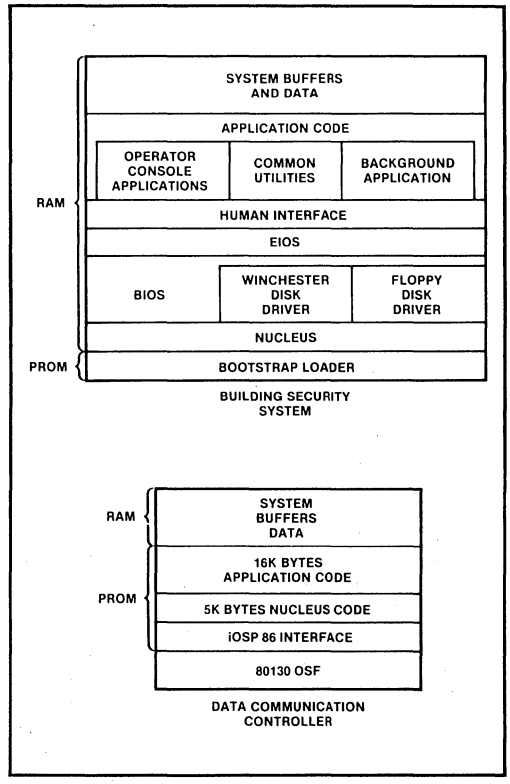


Figure 7. Typical iRMX™ 86 Configurations

Most configuration options are selected during system design stages. Others may be selected during system operation. For example, the amount of memory devoted to queues within a Mailbox can be specified at the time the Mailbox is created. Devoting more memory to the Mailbox allows more messages to be transmitted to other tasks without having to degrade system performance to allocate additional memory dynamically.

The chart shown in Table 6 indicates the actual memory size required to support these different configurations of the iRMX 86 System. Systems requiring only Nucleus level functions may require no more than 13K bytes for the Operating System. (Use of the iAPX 86/30 requires only 4K bytes of RAM, 7K bytes of initialization code in EPROM and the 16K bytes of code in the 80130.) Other applications, needing I/O management functions, may select portions of additional layers that fit their needs and size constraints.

This configurability also applies to the Terminal Handler, Dynamic Debugger, and System Debugger. The Terminal Handler provides a serial terminal interface in a system that otherwise doesn't need an I/O system. Either one of the debuggers need to be included only as debugging tools (usually only during system development).

Order Number: 210885-002

Table 6. iRMX™ 86 Configuration Size Chart

System Layer	Min. ROMable Size	Max. Size	Data Size
Bootstrap Loader	1K	1.5K	6K*
Nucleus	10.5K	24K	2K
BIOS	26K	78K	1K
Application Loader	4K	10K	2K
EIOS	10.5K	12.5K	1K
Human Interface	22K	22K	15K
UDI	8K	8K	0
Terminal Handler	3K	3K	0.3K
System Debugger	20K	20K	1K
Dynamic Debugger	28.5K	28.5K	1K
Human Interface Commands			116K
Interactive Configuration Utility			308K

* Usable by System after bootloading.

MULTI-PROCESSING

The resources provided by a single processor are often not enough to perform certain functions. With the standard interfaces provided by the iMMX 800 MULTIBUS Message Exchange package, the iRMX 86 Operating System supports a loosely-coupled multi-processing environment. Task running on one processor may communicate with tasks running on other processors, even if they operate under different operating systems. The iMMX 800 software is capable of sending messages over the MULTIBUS to tasks operating under either the iRMX 88 Executive, or the iRMX 86 Operating System. Using this message exchange mechanism, applications may increase their system performance quite easily, improve overall interrupt response, gain access to the iSBC® 550 Ethernet Controller, and leave room for future product enhancements.

MULTI-USER ACCESS

Many real-time systems must provide a variety of users access to system control functions and collected data. The iRMX 86 System provides easy-to-use support for applications to access multiple terminals. It also enables multiple and different users to access different applications concurrently.

Figure 8 illustrates a typical iRMX 86 application simultaneously supporting multi-terminal data collection and real-time environments. Shown is a group of terminals used by machinists on a shop floor to communicate with a job management program, a building security system that constantly monitors energy usage requirements, a system operator console capable of accessing all system functions, and a group of terminals in the Production Engineering department used to monitor job costs while developing new device control specifications instructions. The iSBC 544 Intelligent Terminal Interface supports multiple user terminals without degrading system performance to handle character I/O.

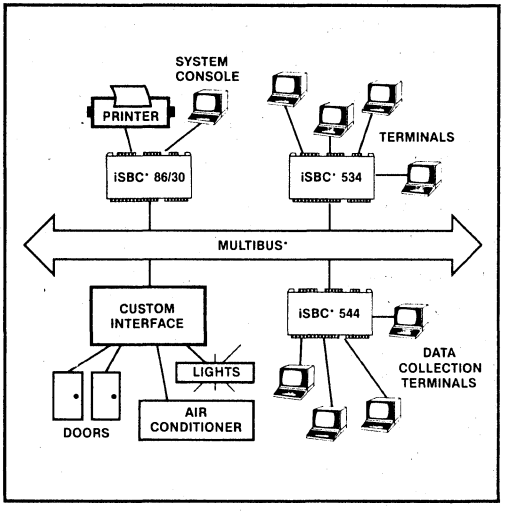


Figure 8. Multi-Terminal and Multi-User Real-Time System

EXTENDABILITY

The iRMX 86 Operating System provides three means of extensions. This extendability is essential for support of OEM and volume end user value added features. This ability is provided by: user-defined operating system calls, user-defined objects (similar to Jobs, Tasks, etc.), and the ability to add functions later in the product life cycle. The modular, layered structure of the System easily facilitates later additions to iRMX 86 applications. User-defined objects are supported by the functions listed in Table 7.

Using standard iRMX 86 system calls, users may define custom objects, enabling applications to easily manipulate commonly used structures as if they were part of the original operating system.

Table 7. User Extension System Calls

System Call	Function Performed
RQ$CREATE$COMPOSITE	Creates a custom object built of previously defined objects.
RQ$DELETE$COMPOSITE	Deletes the custom object, but not the various objects from which it was built.
RQ$INSPECT$COMPOSITE	Returns a list of Token Identifiers for the component objects from which the specified composite object is built.
RQ$ALTER$COMPOSITE	Replaces a component object of a composite object.
RQ$CREATE$EXTENSION	Creates a new type of object and assigns a mailbox used for collecting these objects when they are deleted.
RQ$DELETE$EXTENSION	Deletes an extension definition.

EXCEPTION HANDLING

The System includes predefined exception handlers for typical I/O and parameter error conditions. The error handling mechanism is both configurable and extendable.

SUPPORT OF STANDARDS

The iRMX 86 Operating System supports the many hardware and software standards needed by most application systems to ensure that commonly available hardware and software packages may be interfaced with a minimum of cost and effort. The iRMX 86 System supports the iSBC family of products built on the Intel MULTIBUS (IEEE Standard 796), and a number of standard software interfaces such as the UDI and the common device driver interface (See Figure 9). The procedural interfaces of the UDI are listed in Table 9.

The Operating System includes support for the proposed IEEE 80-bit extended real-variable format of the 8087 Numeric Data Processor, and the IEEE 796 (MULTIBUS) hardware interface. Other standards such as the iMMX 800 MULTIBUS Message Exchange, and an Ethernet communication interface are supported by optional software packages available to run on the iRMX 86 System.

SPECTRUM OF CPU PERFORMANCE

The iRMX 86 Operating System supports a broad range of Intel processors. In addition to support for iAPX 86 and 88 based systems, the iRMX 86 system has been enhanced to support iAPX 186, 188, and 286 (Real Address Mode)-based Systems. This new support enables the user to take advantage of the faster speed and higher performance of Intel's 286 based microprocessors such as the iSBC 286/10 single board computer. By choosing the appropriate CPU, designers can choose from a wide range of performance options, without having to change application software.

COMPONENT LEVEL SUPPORT

The iRMX 86 System may be tailored to support specific hardware configurations. In addition to system memory,

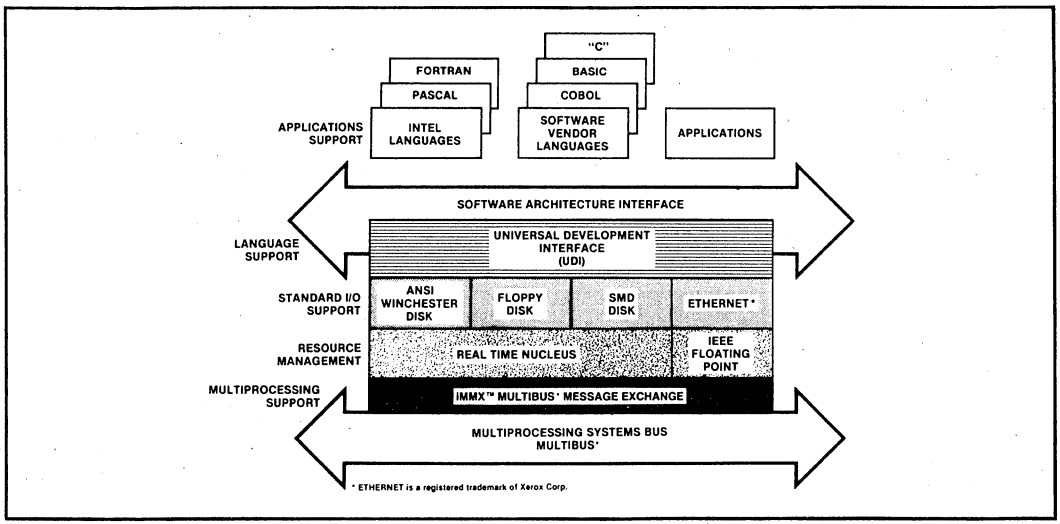


Figure 9. iRMX™ 86 Standard Interfaces

Order Number: 210885-002

only an iAPX 86, iAPX 88, iAPX 186, iAPX 188, or iAPX 286 microprocessor, an 8259A Programmable Interrupt Controller (PIC), and either an 8253, 8274, or 82530 Programmable Interval Timer (PIT) are required as follows:

- iAPX 86 and iAPX 88 systems need either:
  - 8253 PIT and 8259A PIC (master) or
  - 80130 firmware (PIC is master)
- iAPX 186 and iAPX 188 systems where 186 PIC is slave, needs either:
  - 8253 PIT and 8259A PIC (master) or
  - 80130 firmware (PIC is master)

  where 186 PIC is master:
  - Uses 186 PIT for the system clock; no external PIT is needed
  - Can use either
    186 PIC (master) only or
    8259A/80130 PIC (slave)
- iAPX 286 systems need
  - 8253 PIT and 8259A PIC.

Alternatively, the iRMX 86 Operating System may be used in conjunction with the 80130 Operating System Firmware Component that not only provides these hardware functions, but eliminates the need for approximately 16K bytes of the iRMX 86 Nucleus code (see Figure 7). For systems requiring extended mathematics capability, an 8087 or 80287 Numeric Data Processor may be added to perform these functions up to 100 times faster than equivalent software. For applications servicing more than 8 interrupt sources, additional 8259A's may be configured as slave controllers.

BOARD LEVEL SUPPORT

The iRMX 86 Operating System includes device drivers to support a broad range of MULTIBUS device controllers. The particular boards and types of devices supported are listed in Table 8. The device controllers all adhere to industry standard electrical and functional interfaces.

In addition to the on-CPU board terminal drivers, the iRMX 86 BIOS includes two iSBC board-level device drivers to support multiple terminal interfaces:

The iSBC 544 Intelligent Four-Channel Terminal Interface Device Driver provides support for multiple controllers each supporting up to four standard RS232 terminals. The iSBC 544 driver takes advantage of an on-board 8085 processor to greatly reduce the system processor time required for terminal I/O by locally managing input and output buffers. The iSBC 544 firmware provided with the operating system can offload the system CPU by as much as 75% when doing character outputting.

The iSBC 534 Four-Channel USART Controller Device Driver also provides support for multiple controller boards each supporting up to four standard RS232 terminals.

The new RAM disk feature in iRMX 86 makes a portion of the memory address space look like a disk drive to the I/O system.

Table 8. Supported Devices

iSBC® Device Controller	Description
iSBC® 86,88	Serial Port to CRT, Parallel Port to Centronics-type Printer, Interval Timer and Interrupt Controller.
iSBC® 186/03	Small Computer System Interface (SCSI) Supporting All Random Access "Extended Standard" SCSI/SASI hard disk controllers.
iSBC® 204	Single Density Diskette.
iSBC® 206	Cartridge-Type Hard Disk.
iSBC® 208	Single & Double Density, Single & Double Sided, 8" & 5.25" Diskettes.
iSBC® 215(G)	Standard Winchester Disks.
iSBX® 218	Single or Double density, Single or double sided, 8-inch diskettes (when used on an iSBC 215(G)).
iSBX® 218A	Single or Double Density, Single or Double Sided, 8" & 5.25" Diskette (when used on an iSBC 215G Winchester Controller).
iSBC® 220	Standard Storage Module Board.
iSBX® 251	Bubble Memory Multimodule Board.
iSBC® 254(S)	Bubble Memory Board.
iSBX® 351	1-Channel Serial Port to CRTs, Modems.
iSBC® 534,544	4-Channel Serial Ports to CRTs, Modems.
iSBX™ 270	Black and White CRTs and full ASCII keyboards.

NOTE: (G) = optional iSBC 215, iSBC 215B, or iSBC 215G
(S) = optional iSBC 254 or iSBC 254S

Development Environment Features

The iRMX 86 Operating System supports the efficient utilization of programming time by providing important tools for program development. Some of the tools necessary to develop and debug real-time systems are included with the Operating System. Others, such as language compilers, are available from Intel and from leading Independent Software Vendors.

LANGUAGES

The iRMX 86 Operating System supports 31 standard system calls known as the Universal Development Interface (UDI). Figure 9 shows the iRMX 86 standard interfaces to many compilers and language translators, including the iAPX 86 and 88 Macro Assembler; the PASCAL 86/88, PL/M 86/88, FORTRAN 86/88 and C86 compilers available from Intel. Also included are other

Intel development tools, language translators and utilities available from other vendors. Any application that ran on the iRMX 86 Release 5 Universal Runtime Interface (URI) will run on the iRMX 86 Release 6 UDI. The full set of UDI calls (which includes the URI system calls) is required to run a compiler.

These standard software interfaces (the UDI) ensure that users of the iRMX 86 Operating System may transport their applications to future releases of the iRMX 86 Operating System and other Intel and independent vendor software products. The calls available in the UDI are shown in Table 9.

Table 9. UDI System Calls

System Call	Function Performed
Memory Management:	
DQ$ALLOCATE	Creates a Segment of a specified size.
DQ$FREE	Returns the specified segment to the System.
DQGETSIZE*	Returns the size of the specified Segment.
DQ$RESERVE$IO$MEMORY*	Reserves memory to OPEN and ATTACH files.
File Management:	
DQ$ATTACH	Creates a Connection to a specified file.
DQ$CHANGE$ACCESS*	Changes the user access rights associated with a file or directory.
DQ$CHANGE$EXTENSION	Changes the extension of a file name in memory.
DQ$CLOSE	Closes the specified file Connection.
DQ$CREATE	Creates a Named File.
DQ$DELETE	Deletes a Named File.
DQ$DETACH	Closes a Named File and deletes its Connection.
DQ$OPEN	Opens a file for a particular type of access.
DQGETCONNECTION$STATUS*	Returns the current status of the specified file Connection
DQ$FILE$INFO*	Returns data about a file Connection.
DQ$READ	Reads the next sequence of bytes from a file.
DQ$RENAME*	Renames the specified Named File.
DQ$SEEK	Moves the position pointer of a file.
DQ$TRUNCATE	Truncates a file.
DQ$WRITE	Writes a sequence of bytes to a file.
Process Management:	
DQ$EXIT	Exits from the current application job.
DQ$OVERLAY*	Causes the specified overlay to be loaded.
DQ$SPECIAL	Performs special I/O related functions on terminals with special control features.
DQ$TRAP$CC	Captures control when CNTRL/C is typed.
Exception Handling:	
DQGETEXCEPTION$HANDLER	Returns a pointer to the program currently being used to process errors.
DQ$DECODE$EXCEPTION	Returns a short description of the specified error code.
DQ$TRAP$EXCEPTION	Identifies a custom exception processing program for a particular type of error.
Application Assistance:	
DQ$DECODE$TIME	Returns system time and date in binary and ASCII character format.
DQGETARGUMENT*	Returns the next argument from the character string used to invoke the application program.
DQGETSYSTEM$ID*	Returns the name of the underlying operating system supporting the UDI.
DQGETTIME*	Returns the current time of day as kept by the underlying operating system.
DQ$SWITCH$BUFFER	Selects a new buffer from which to process commands.

* Calls available only through the UDI.

The high performance of the iRMX 86 Operating System enhances the throughput of compilers and other development utilities. Table 10 indicates the average performance of typical development environment functions operating in the same configuration described in Figure 4.

Table 10. Development Environment Performance

Function	Average Execution Time
Directory Command (S Format with 25 files)	5.3 sec
Load the COPY Command	1.2 sec
Copy a 1K Byte File (Winchester to Winchester)	1.0 sec
Copy a 16K Byte File	1.7 sec
Copy a 64K Byte File	3.9 sec
Copy a 1K Byte File (Winchester to Diskette)	1.4 sec
Compile PL/M 86	393 lpm
Compile PASCAL 86 Program	453 lpm

TOOLS

Certain tools are necessary for the development of microcomputer applications. The iRMX 86 Human Interface includes many of these tools as non-resident commands. They can be included on the system disk of a application system, and brought into memory when needed to perform functions as listed in Table 11.

Table 11. Major Human Interface Utilities

Command	Function
BACKUP	Copy directories and files from one device to another.
COPY	Copy one or more files to one or more destination files.
CREATEDIR	Create a directory file to store the names of other files.
DIR	List the names, sizes, owners, etc. of the files contained in a directory.
ATTACHFILE	Give a logical name to a specified location in a file directory tree.
PERMIT	Grant or rescind user access to a file.
RENAME	Change the name of a file.
SUBMIT	Start the processing of a series of commands stored in a file.
SUPER	Change operator's ID to that of the System Manager with global access rights and privileges.

Table 11. Major Human Interface Utilities (Con.t.)

Command	Function
TIME	Set the system time-of-day clock.
VERIFY	Verify the structure of an iRMX™ 86 Named File volume, and check for possible disk data errors.

INTERACTIVE CONFIGURATION UTILITY

The iRMX 86 Operating System is designed to provide OEMs the ability to configure for specific system hardware and software requirements. The Interactive Configuration Utility (ICU) builds iRMX 86 configurations by asking appropriate questions and making reasonable assumptions. It runs on either an Intellec® Series III development system or iRMX 86 development system that includes a hard disk and the UDI. Table 12 lists the hardware and support software requirements of different iRMX 86 development system environments.

Table 12. iRMX™ Development Environment

Intellec® Series III: MDS 313 PL/M 86/88 Compiler One hard disk and one diskette drive
iRMX™ 86 Development System iRMX™ 860 ASM 86 Assembler and Utilities iRMX™863 PL/M 86/88 Compiler iSDM 86 or 286 System Debug Monitor 512K Bytes of RAM 5M Byte On-Line Storage and one double-density diskette drive
SYSTEM 86/300 or 286/300 Series Microcomputer System Basic configuration

Figure 10 shows one of the many screens displayed during the process of defining a configuration. It shows the abbreviations for each choice on the left, a more complete description with the range of possible answers in the center, and the current (sometimes default) choice on the right. The bottom of the screen shows three changes made by the operator (lower case lettering), and a request for help on the Exception Mode question. In response to a request for help, the ICU displays an additional screen outlining possible choices and some overall system effects.

The ICU requests only information required as a result of previous choices. For example, if no Extended I/O System functions are required, the ICU will not ask any further questions about the EIOS. Once a configuration session is complete, the operator may save all the information in a file. Later when small changes are necessary, this file can be modified. A completely new session is not required.

```
Nucleus
(ASC)   All Sys Calls [Yes/No]                          Yes
(PV)    Parameter Validation [Yes/No]                   Yes
(ROD)   Root Object Directory Size [0 - 0FF0h]          0014H
(MTS)   Minimum Transfer Size [0 - 0FFFFH]              0040H
(DEH)   Default Exception Handler [Yes/No/Deb/Use]      Yes
(NEH)   Name of Ex Handler Object Module [1 - 32chs]
(EM)    Exception Mode [Never/Program/Environ/All]      Never
(NR)    Nucleus in ROM [Yes/No]                         No

Enter Changes [Abbreviations ?/ = new-value] : ASC = N
:pv = no
:rod = 48
:em ?
```

Figure 10. ICU Screen for iRMX™ 86 Nucleus

REAL-TIME DEBUGGING TOOLS

The iRMX 86 Operating System supports three distinct debugging environments: Static, Dynamic, and Post-Mortem. While the iRMX 86 Operating System does support a multi-user Human Interface, these real-time debugging aids are usually most useful in a single-user environment where modifications made to the system cannot affect other users.

System Debugger

The static debugging aid is the iRMX 86 System Debugger. This debugger is an extension of the iSDM 86 and the iSDM 286 System Debug Monitors. The System Debugger provides static debugging facilities when the system hangs or crashes, when the Nucleus is inadvertently overwritten or destroyed, or when synchronization requirements prevent the debugging of certain tasks. The System Debugger stops the system and allow you to examine the state of the system at that instant, and allows you to:

— Identify and interpret iRMX 86 system calls.

— Display information about iRMX 86 objects.

— Examine a task's stack to determine system call history.

iRMX™ 86 Dynamic Debugger

The iRMX 86 Dynamic Debugger runs as part of an iRMX 86 application. It may be used at any time during program development, or may be integrated into an OEM system to aid in the discovery of latent errors. The Dynamic Debugger can be used to search for errors in any task, even while the other tasks in the system are running. The iRMX 86 Dynamic Debugger communicates with the developer via a terminal handler that supports full line editing.

System Crash/Dump Analyzer

The often difficult job of debugging real-time applications is made much simpler with the System Crash/Dump Analyzer. The analyzer allows program developers to record system memory for later analysis even if the system has halted. This analysis lists such vital information as which jobs have active tasks, which system queues contain which tasks, and what segments contain which data.

PARAMETER VALIDATION

Some iRMX 86 System Calls require parameters that may change during the course of developing iRMX 86 applications. The iRMX 86 Operating System includes an optional set of routines to validate these parameters to ensure that correct numeric values are used and that correct object types are used where the System expects to manipulate an object. For systems based only on the iRMX 86 Nucleus, these routines may be removed to improve the performance and code size of the System once the development phase is completed.

START-UP SYSTEMS

Two ready-to-run, multi-user start-up systems are included in the iRMX 86 Operating System package. These iRMX 86 start-up systems are fully configured, multi-user iRMX 86 Operating Systems ready to be loaded into memory by the Bootstrap Loader. Both start-up systems are configured to include all of the system calls for each layer and most of the features provided by iRMX 86. iRMX start-up systems include UDI support so that users may run languages such as PL/M-86, Pascal, FORTRAN, and software packages from independent vendors.

The start-up system for the iAPX 86 processor is configured for Intel SYSTEM 86/300 Series microcomputers with a minimum of 384K bytes of RAM. The following devices are supported.

- iSBC 215/iSBX 218 or iSBC 215G/iSBX 218A
- iSBC 254(S)
- Line Printer
- 8251A Terminal Driver
- iSBC 544 Terminal Driver

The start-up system for the iAPX 286 processor is configured for Intel SYSTEM 286/300 Series microcomputers with a minimum of 512K bytes and a maximum of 896K bytes of RAM. The following devices are supported.

- iSBC 208
- iSBC 215/iSBX 218 or iSBC 215G/iSBX 218A
- iSBC 254(S)
- Line Printer for iSBC 286/10
- 8274 Terminal Driver
- iSBC 544 Terminal Driver

Either system will run without hardware or software configuration changes and can be reconfigured on a standard system with at least 512K bytes of RAM. Definition files are also included for iSBC 186/03, 186/51 and 188/48 configurations.

This start-up system may be used to run the ICU (if a Winchester disk is attached to the system) to develop custom configurations such as those pictured in Figure 8. As shipped, the Human Interface supports a single user terminal. However, the Start-up System terminal configuration file may be altered easily to support from two to five users.

Order Number: 210885-002

SPECIFICATIONS

Supported Software Products

iRMX 860	iRMX 86 Development Utilities Package, including the iAPX 86 and 88 Linker, Locater, Macro Assembler, Librarian, and the iRMX 86 Editor.
iRMX 861	PASCAL 86/88 Compiler
iRMX 862	FORTRAN 86/88 Compiler
iRMX 863	PL/M 86/88 Compiler
iRMX 864	TX Screen-oriented Editor
iMMX 800	MULTIBUS Message Exchange software package for iRMX 86, and 88 application systems
iOSP 86	Support Package for iAPX 86/30, 88/30, 186/30, and 188/30 Operating System Processors
iRMX PSCOPE 86	High Level Language Debugger

Supported Hardware Products

COMPONENTS

iAPX 86 and 88 Microprocessors

iAPX 186 and 188 Microprocessors

iAPX 286 Microprocessors (Real Address Mode only)

8087 Numeric Data Processor Extension

80287 Numeric Data Processor Extension

iAPX 86/30 (80130) Operating System Firmware Component

8253 and 8254 Programmable Interval Timers

8259A Programmable Interrupt Controller

8251A USART Terminal Controller

8255 Programmable Parallel Interface

8274 Terminal Controller

82530 Serial Communications Controller

iSBC® MULTIBUS BOARD AND SYSTEM PRODUCTS

iSBC 86/12A, 86/05, 86/14, 86/30, 86/35, 88/25, and 88/40 Single Board Computers

iSBC 186/03 Single Board Computer

iSBC 186/51 Ethernet Controller

iSBC 188/48 Communications Controller

iSBC 286/10 Single Board Computer(Real Address Mode only)

iSBC 204 Diskette Controller

iSBC 206 Hard Disk Controller

iSBC 208 Diskette Controller

iSBC 215(G) Winchester Disk Controller

iSBX 218(A) Flexible Diskette Multi-Module Controller

iSBC 220 SMD Disk Hard Controller

iSBC 254(S) Bubble Memory System

iSBC 534 4-Channel Terminal Interface

iSBC 544 Intelligent 4-Channel Terminal Interface and Controller

iSBX 251 Bubble Memory Multi-Module

iSBX 350 Parallel Port (Centronics-type Printer Interface)

iSBX 351 Serial Communications Port

iSBX 270 CRT Light Pen and Keyboard Interface

SYSTEM 86/300 Family

SYSTEM 286/300 Family

AVAILABLE LITERATURE

The iRMX 86 Documentation Set is comprised of the following four volumes of reference manuals. Order numbers are associated with these four volumes only.

iRMX 86 INTRODUCTION AND OPERATOR'S REFERENCE MANUAL FOR RELEASE 6
Order Number: 146545-001

Introduction to the iRMX 86 Operating System

iRMX 86 Operator's Manual

iRMX 86 Disk Verification Utility Reference Manual

iRMX 86 PROGRAMMERS REFERENCE MANUAL FOR RELEASE 6, PART I
Order Number: 146546-001

iRMX 86 Nucleus Reference Manual

iRMX 86 Basic I/O System Reference Manual

iRMX 86 Extended I/O System Reference Manual

iRMX 86 PROGRAMMERS'S REFERENCE MANUAL FOR RELEASE 6, PART II
Order Number: 146547-001

iRMX 86 Application Loader Reference Manual

iRMX 86 Human Interface Reference Manual

iRMX 86 Universal Development Interface Reference Manual

Guide to Writing Device Drivers for iRMX 86 and iRMX 88 I/O Systems

iRMX 86 Programming Techniques

iRMX 86 Terminal Handler Reference Manual

iRMX 86 Debugger Reference Manual

iRMX 86 System Debugger Reference Manual

iRMX 86 Crash Analyzer Reference Manual

iRMX 86 Bootstrap Loader Reference Manual

iRMX 86 INSTALLATION AND CONFIGURATION GUIDE FOR RELEASE 6
Order Number: 146548-001

iRMX 86 Installation Guide

iRMX 86 Configuration Guide

Master Index for Release 6 of the iRMX 86 Operating System

Application Notes

Ap Note 130 — Using Operating System Processors to Simplify Microcomputer Designs. (Order Number: 230786-001

Ap Note 174 — Optimizing the iRMX 86 Operating System Performance on System 86/310 and System 86/330 (Order Number: 230990-001)

Training Courses

The iRMX 86 Operating System

Customer Seminars

Contact local Intel Sales Office for details on available video-tape and slide presentations.

ORDERING INFORMATION

The iRMX 86 Operating System is available under a number of different licensing options as noted here. Source listings are available on microfiche. Reconfigurable object libraries are provided on double density ISIS-formatted diskettes or on either double density, single sided iRMX 86-formatted 8″ diskettes, or double density, double sided, 5.25″ diskettes. ISIS-format diskettes may be used on Intel Intellec Development Systems. The iRMX 86-format may be used on any iRMX 86-based system supporting the appropriate compilers and development environment.

The OEM license options listed here allow users to incorporate the iRMX 86 Operating System into their applications. Each use requires payment of an Incorporation Fee.

ORDER CODE	DESCRIPTION
iRMX 86 KIT BRO:	Double density, single-sided 8″ ISIS format OEM license
iRMX 86 KIT ERO:	Double density, single sided 8″ iRMX 86-Format OEM license for use on iRMX 86-based environments.
iRMX 86 KIT JRO:	Double density, double sided 5.25″ iRMX 86-Format OEM license for use on iRMX 86-based environments.

Other licensing options include prepayment of all future incorporation fees, single use rights for a single machine, use at a second development site, one year update service extensions, the right to make copies for additional development systems, and source listing materials.

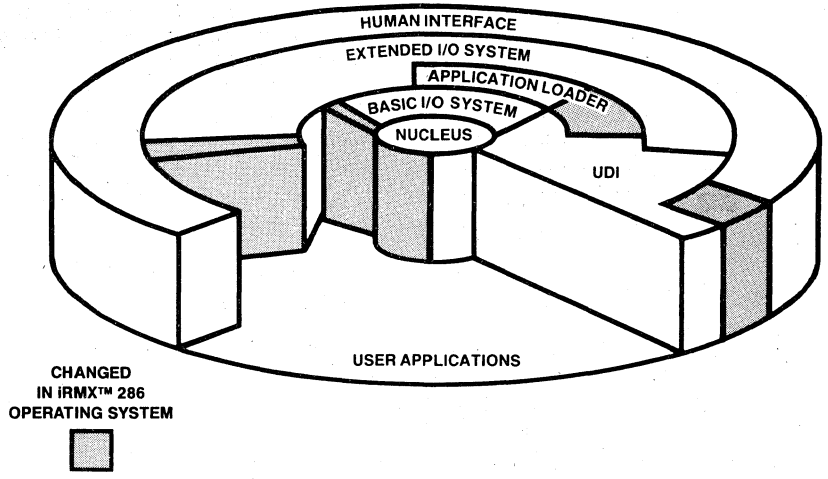
Each option includes 90 days of support service that provides the quarterly iRMX 86 Technical Report, Software Problem Report Service, and copies of System Updates that occur during this period. Except for source listings, all initial licenses include a complete set of iRMX 86 Documentation.

As with all Intel software, purchase of any of these options requires the execution of a standard Intel Master Software License. The specific rights granted to users depends on the specific option and the License signed.

iRMX™ 286 OPERATING SYSTEM

- **Real-time processor management for time-critical iAPX 286 applications**
- **16 Megabytes of memory addressable**
- **Support of iAPX 286 processor in native mode, using 286 Object Module Format (OMF)**
- **On-target system development with Universal Development Interface (UDI)**
- **Configurable system size and function for diverse application requirements**
- **All iRMX™ 286 code can be (P)ROMed to support totally solid state designs**
- **Configured system for the iAPX processor in Intel integrated system products (SYSTEM 286/300)**
- **Multi-terminal support with multi-user human interface**
- **Device drivers included for industry standard MULTIBUS® peripheral controllers**
- **Complete support of 80287 processor extension**
- **Powerful utilities for interactive configuration and debugging**

The iRMX™ 286 Operating System provides a full featured, easy-to-use software system which takes advantage of the native mode of the iAPX 286 processor to provide extended addressing and greater reliability than the iRMX 86 Operating System. The iRMX 286 Operating System is a real-time, multi-tasking and multi-programming system designed to manage and extend the resources of iSBC® 286 Single Board Computers, as well as other iAPX 286-based microcomputers. IRMX 286 Operating System functions are also fully integrated into the SYSTEM 286/300 Family of Microcomputer Systems.



iRMX™ 286 VLSI Operating System

SEPTEMBER, 1984
ORDER NUMBER: 280082-001

The iRMX 286 Operating System is a complete set of system software modules that provide the resource management functions very similar to those currently available to OEMs in Release 6 of the iRMX 86 Operating System. For a complete description of these functions, please refer to the iRMX 86 Release 6 Data Sheet (order number 210885-002).

This advanced information sheet describes the major features and new concepts provided by the iRMX 286 Operating System. These features enable OEMs to take advantage of the extended memory and greater reliability provided by the iAPX 286 microprocessor.

FUNCTIONAL DESCRIPTION

The iRMX 286 Operating System provides greater addressability than the iRMX 86 Operating System, allowing up to 16 megabytes of memory to be used in a system. Added reliability is obtained in the iRMX 286 Operating System because of the presence of additional hardware traps, the ability to mark code segments executable-only, and indirectly accessing memory through descriptor table entries.

The iAPX 286 processor is a general purpose chip and can support a variety of operating systems. Consequently, each operating system will use only some of the functions of the iAPX 286 chip. The iRMX 286 Operating System does not use the four ring model of protection available from the processor.

Because the iRMX 286 Operating System is based on the iRMX 86 Operating System, application source code will be mostly compatible between the operating systems. One system call in the nucleus was replaced (RQSETOS$EXTENSION); all other system calls can be used without modifications. Source code simply needs to be recompiled with the iAPX 286 compilers and then built or bound with iAPX 286 development utilities.

To take advantage of the increased address space provided with the iRMX 286 Operating System, new system calls that can specify up to 16 megabytes of memory can be used. Multiple applications, each based on system calls that have one megabyte per job memory limits, can also be loaded into the increased address space.

In the iRMX 286 Operating System, memory is accessed indirectly through entries in the descriptor table. Indirect addressing prevents applications from directly writing to memory, which improves system integrity.

Devices used with the iRMX 286 Operating System must be able to access 16 Megabytes, or the device must be an iSBX™ MULTIMODULE ™ board. The device drivers provided with the iRMX 286 Operating System include:

- An 8274 terminal device driver, to support the 8274 for two channel Asynchronous support of the iSBC® 286/10 board in RS232 mode.
- Support for the iSBC 215G board to control standard Winchester disks.
- Support for the iSBX™ 218A Flexible Disk Controller, both on the iSBC 215G Controller and on the CPU.
- The iSBX 251 Bubble Memory MULTIMODULE Board.
- The iSBX 350 Line Printer MULTIMODULE Board.
- The iSBX 351 1-Channel Serial Port to CRTs or Modems.

SPECIFICATIONS

Supported Software Products

iRMX 286 Development Utilities Package, including the iAPX 286 Builder, Binder, Macro Assembler, Librarian, Overlay Generator, and Mapper.
FORTRAN 286 Compiler
PL/M 286 Compiler
AEDIT Screen-oriented Editor

Supported Hardware Products

COMPONENTS

iAPX 286 Microprocessors (Native Mode)
80287 Numeric Data Processor Extension
8253 and 8254 Programmable Interval Timers
8259A Programmable Interrupt Controller
8251A USART Terminal Controller on iSBX 351 MULTIMODULE board
8255 Programmable Parallel Interface
8274 Terminal Controller

iSBC® MULTIBUS® BOARD AND SYSTEM PRODUCTS

iSBC 286/10 Single Board Computer (Native Mode)
iSBC 215G Winchester Disk Controller
iSBX 218A Flexible Diskette MULTIMODULE board Controller
iSBX 251 Bubble Memory MULTIMODULE board
iSBX 350 Parallel Port (Centronics-type Printer Interface)
iSBX 351 Serial Communications Port
SYSTEM 286/310-17

LITERATURE

The iRMX 286 Documentation Set will be comprised of the four volumes of the iRMX 86 Operating System, described in the iRMX 86 Data Sheet, plus an additional volume, the iRMX 286 Reference Manual for Release 1.

ORDERING INFORMATION

The iRMX 286 Operating System will be available under a number of different licensing options. Source listings will be available on human readable microfiche. Reconfigurable object libraries will be provided on either double density, single sided iRMX 286-formatted 8" diskettes (Media E), or double density, double sided, 5.25" diskettes (Media J). The iRMX 286-format may be used on any iRMX 286-based system supporting the appropriate compilers and development environment.

The OEM license options allow users to incorporate the iRMX 286 Operating System into their applications. Each use requires payment of an Incorporation Fee.

intel

iRMX™ 51
REAL-TIME MULTITASKING EXECUTIVE

- **Software tool for family of 8051 microcontroller based applications**
- **Real-time, multitasking executive**
- **Supports remote task communication**
- **Small — 2.2K Bytes**
- **Reliable**
- **Simple user interface**
- **Compatible with BITBUS™/Distributed Control Modules (iDCM) product line: iSBX™ 344 & iRCB 44/10 boards**

The iRMX™ 51 Executive is a compact, easy to use, software tool for development and implementation of applications built on the high performance 8-bit family of 8051 microcontrollers. A few members of this expansive family are the 8051, 8044, and 8052 microcontrollers. Like the 8051 family, the iRMX 51 Executive incorporates many features that make it exceptionally well suited for real-time control applications requiring manipulation and scheduling of more than one job, and fast response to external stimuli.

The 8051 microcontroller family is the family of choice for applications such as: data acquisition and monitoring, process control, robotics, and machine control. Using the iRMX 51 Executive for a foundation can significantly reduce applications development time. Also, the iRMX 51 Executive fully supports Intel's BITBUS™ microcontroller interconnect expressly designed for reliable high performance real-time control.

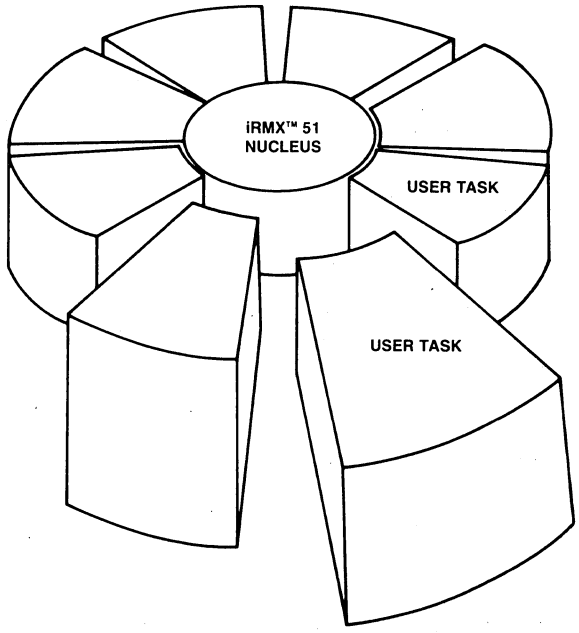


Figure 1. Structure Diagram

ARCHITECTURE

Real-time and Multitasking

Real-time control applications must be responsive to the external environment and typically involve the execution of more than one function (task or set of tasks) in response to different external stimuli. Control of an industrial drying process is an example. This process could require monitoring of multiple temperatures and humidity; control of fans, heaters, and motors that must respond accordingly to a variety of inputs. The iRMX 51 Executive fully supports applications requiring response to stimuli as they occur ie. in real-time. This real-time response is supported for multiple tasks often needed to implement a control application.

Some of the facilities precisely tailored for development and implementation of real-time control application systems provided by the iRMX 51 Executive are: task management, interrupt handling, message passing, and when intergrated with communications support, message passing with different microcontrollers. Also, the iRMX 51 Executive is driven by events: interrupts, timers, and messages ensuring the application system always responds to the environment appropriately.

Task Management

A task is a program defined by the user to execute a particular control function or functions. Multiple programs or tasks may be required to implement a particular function such as 'controlling Heater 1'. The iRMX 51 Executive recognizes three different task states as one of the mechanisms to accomplish scheduling of up to eight tasks. Figure 2 illustrates the different task states and their relationship to one another.

The scheduling of tasks is priority based. The user can prioritize tasks to reflect their relative importance within the overall control scheme. For instance, if Heater 1 must go off line prior to Heater 2 then the task associated with Heater 1 shutdown could be assigned a higher priority ensuring the correct shutdown sequence. The RQ WAIT system call is also a scheduling tool. In this example the task implementing Heater 2 shutdown could include an instruction to wait for completion of the task that implements Heater 1 shutdown.

The iRMX 51 Executive allows for PREEMPTION of a task that is currently being executed. This means that if some external event occurs such as a catastrophic failure of Heater 1, a higher priority task associated with the interrupt, message, or timeout resulting from the failure will preempt the running task. Preemption ensures the emergency will be responded to immediately. This is crucial for real-time control application systems.

Interrupt Handling

The iRMX 51 executive supports sixteen interrupt sources as shown in Table 1. Four of these interrupt sources, excluding timer 0, can be as-

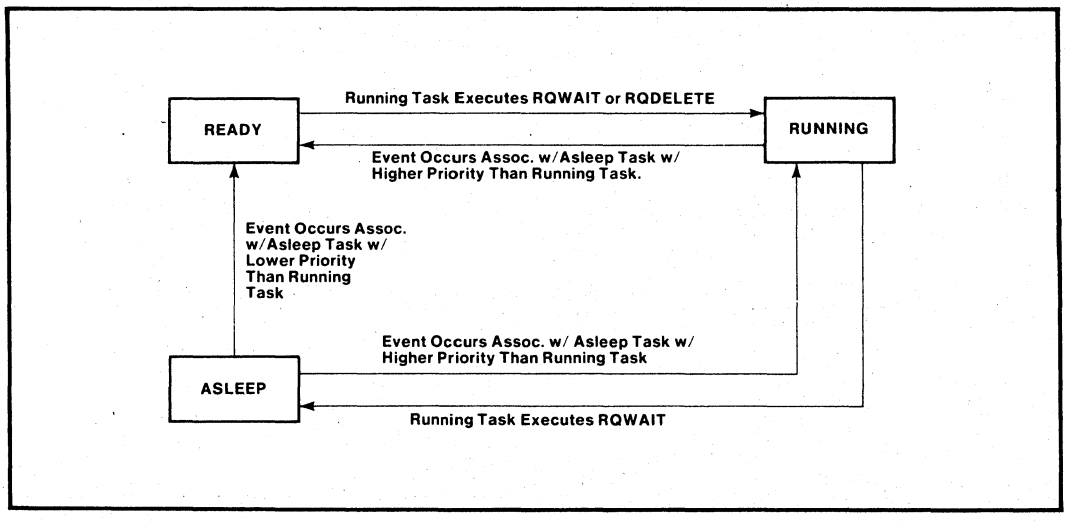


Figure 2. Task State Transition Diagram

signed to a task. When one of the interrupts occurs the task associated with it becomes a running task (if it were the highest priority task in a ready state). In this way, the iRMX 51 Executive responds to a number of internal and external stimuli including time intervals designated by the user.

Table 1. iRMX™ 51 Interrupt Sources

INTERRUPT SOURCE	INTERRUPT NUMBER
External Request 0	00H
Timer 0	01H
External Request 1	02H
Timer 1	03H
Internal Serial Port 1	04H
Reserved	05H
Reserved	06H
Reserved	07H
Reserved	08H
Reserved	09H
Reserved	0AH
Reserved	0BH
Reserved	0CH
Reserved	0DH
Reserved	0EH
Reserved	0FH

Message Passing

The iRMX 51 Executive allows tasks to interface with one another via a simple message passing facility. This message passing facility can be extended to different processors when communications support is integrated within a BITBUS/iDCM system, for example. This facility provides the user with the ability to link different functions or tasks. Linkage between tasks/functions is typically required to support development of complex control applications with multiple sensors (inputs variables) and drivers (output variables). For instance, the industrial drying process might require a dozen temperature inputs, six moisture readings, and control of: three fans, two conveyor motors, a dryer motor, and a pneumatic conveyor. The data gathered from both the temperature and humidity sensors could be processed. Two tasks might be required to gather the data and process it. One task could perform a part of the analysis, then include a pointer to the next task to complete the next part of the analysis . The tasks could continue to move between one another.

REMOTE TASK COMMUNICATION

The iRMX 51 Executive system calls can support communication to tasks on remote controllers. This feature makes the iRMX 51 Executive ideal for applications using distributed architectures. Providing communication support saves significant application development time and allows for more effective use of this time. Intel's iDCM product line combines hardware and software to provide this function.

In an iDCM system, communication between nodes occurs via the BITBUS microcontroller interconnect. The BITBUS microcontroller interconnect is a high performance serial control bus specifically intended for use in applications built on distributed architectures. The iRMX 51 Executive provides BITBUS support.

BITBUS™/iDCM COMPATIBLE

A pre-configured version of the iRMX 51 Executive implements the BITBUS message format and provides all iRMX 51 facilities mentioned previously: task management, interrupt handling, and message passing. This version of the Executive is supplied in firmware on the iDCM Controller with the iDCM hardware products: the iSBX 344 BITBUS Controller MULTIMODULE and the iRCB 44/10 BITBUS Remote Controller boards. It is also supplied on diskette as part of the iRMX 510 iDCM Support Package to ease development of BITBUS systems.

SIMPLE USER INTERFACE

The iRMX 51 Executive's capabilities are utilized through system calls. These interfaces have been defined for ease of use and simplicity. Table 2 includes a listing of these interfaces and their functions. Note tasks may be created at system initialization or run-time using the CREATE TASK call.

Functions such as GET FUNCTION IDS, ALLOCATE/DEALLOCATE BUFFER, and SEND MESSAGE (Messages in the iRMX 51 Executive have a maximum size of 255 bytes.), support communication for distributed architectures. Architectures that define multiple remote stations requiring intelligent and dumb I/O manipulation. The remaining

Table 2. iRMX™ 51 System Interfaces

COMMAND	DESCRIPTION
RQ SEND MESSAGE	Sends a message (a command from the BITBUS master, a response from a slave, or a simple message between tasks on the same BITBUS component) to another task.
RQ WAIT	Waits for an interrupt, an event time-out, a message, or any combination of the three.
RQ CREATE TASK	Causes a new sequence of code to be run as an iRMX 51 task with a specific function identification code and priority.
RQ DELETE TASK	Stops the specified task and removes it from all execution lists.
RQ ALLOCATE	Allocates a fixed-length buffer from the on-chip, scratch-pad RAM for general use, or, in BITBUS applications, for a BITBUS message buffer.
RQ DEALLOCATE	Returns an on-chip buffer to the system.
RQ SET INTERVAL	Set the time interval to be used as a separate event-timer for the task.
RQ ENABLE INTERRUPT	Allow external interrupts to signal the microcontroller.
RQ DISSABLE INTERRUPT	Stops all external interrupts from signaling the microcontroller.
RQ GET FUNCTION ID	Provides a list of the 8 function identification codes representing the tasks currently operating on the microcontroller.

interfaces allow the user to specify the system's response to the external environment — a must for real-time control.

Another feature that eases application development is automatic register bank allocation. The Executive will assign tasks to register banks automatically unless a specific request is made. The iRMX 51 Executive keeps track of the register assignments allowing the user to concentrate on other activities.

The user configures an iRMX 51 system simply by: specifying the initial set of task descriptors and configuration values, and linking the system via the RL 51 Linker and Locator Program with user programs. The nature of the task descriptors allows the user to develop programs, locate them in off-chip ROM, and access them without writing additional code. Programs may be written in ASM 51 or PL/M 51. (Intel's 8051 Software Development Package contains both ASM 51 and RL 51. The iRMX 51 Executive supplies the configuration file and macro defining initial task descriptors.) Figure 3 shows the relationships that exist in the system generation process.

RELIABILE

Real-time control applications require reliability. The nucleus requires about 2K bytes of code space, 40 bytes on-chip RAM, & 218 bytes external RAM. Streamlined code increases performance and reliability, and flexibility is not sacrificed as code may be added to either on-chip or external memory.

The iRMX 51 architecture and simple user interface further enhance reliablity and lower cost. For example, the straightforward structure of the user interfaces, and the transparent nature of the scheduling process contribute to reliability of the overall system by minimizing programming effort. Also, modularity increases reliability of the system and lowers cost by allowing user tasks to be refined independent of the system. In this way, errors are identified earlier and can be easily corrected in each isolated module.

In addition, users can assign tasks a Function ID that allows tracking of the tasks associated with a particular control/monitoring function. This feature reduces maintenance and trouble shooting time thus increasing system run time and decreasing cost.

OPERATING ENVIRONMENT

The iRMX 51 Executive supports applications development based on any member of the high performance 8051 family of microcontrollers. The Executive is available on diskette with user linkable libraries or in the Distributed Control Modules (iDCM) controller preconfigured in on-

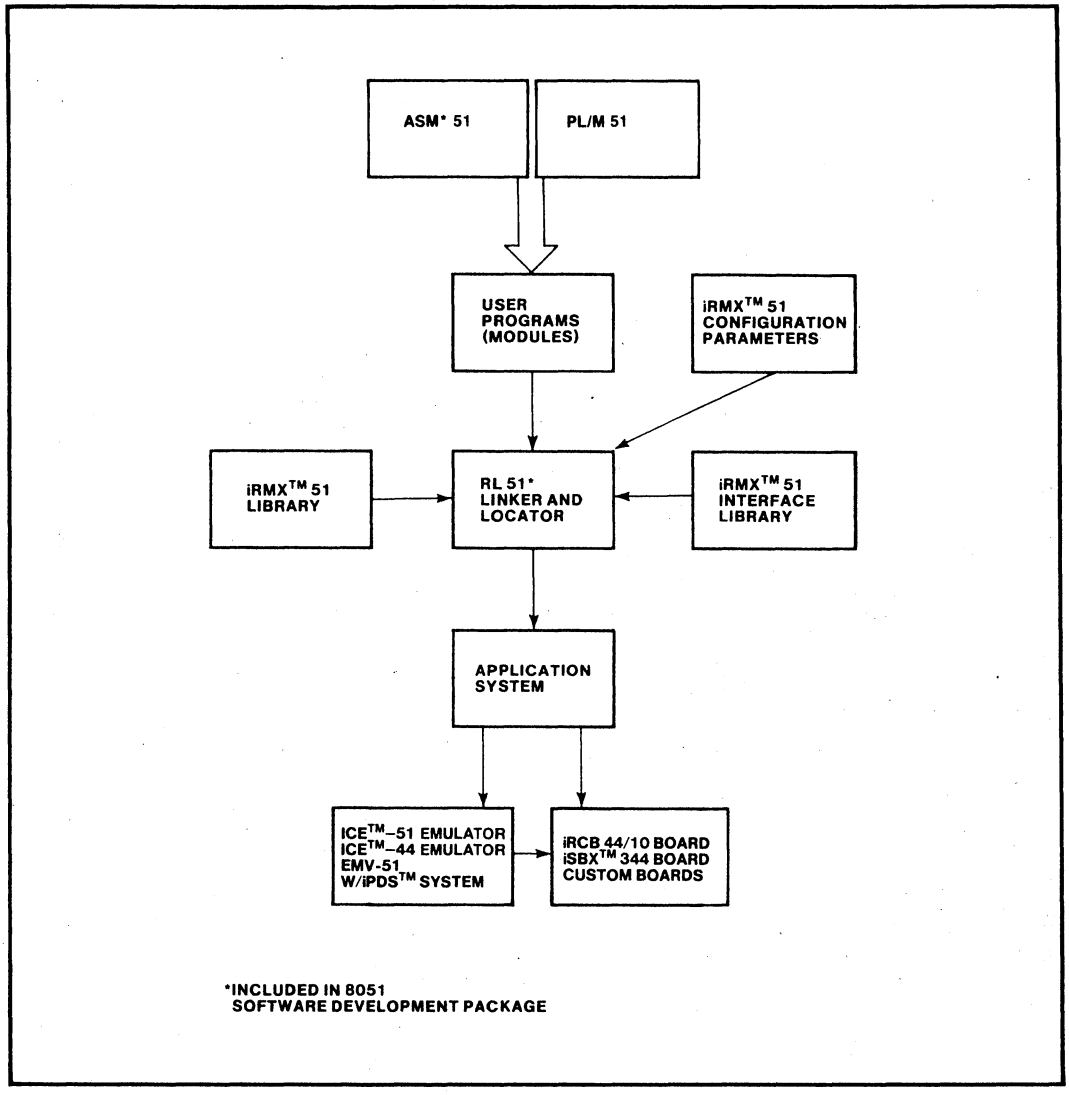


Figure 3. System Generation Process

chip ROM. (The iDCM controller is an 8044 component that consists of an 8051 microcontroller and SDLC controller on one chip with integral firmware.)

When in the iDCM environment (Figure 4), the iRMX 51 Executive can communicate with iRMX based systems like the System 286/310 or ISIS based systems like the Intel Portable Development System (iPDS) by using the iRMX 510 iDCM Support Package.

DEVELOPMENT ENVIRONMENT

Intel provides a complete development environment for the 8051 family of microcontrollers. This environment encompasses iDCM system (BITBUS based) applications also. Software development support consists of: the 8051 Software Development Package, and the iRMX 510 iDCM Support Package. Hardware tools consist of a variety of In Circuit Emulators (ICE), Intel's Portable Development System (iPDS) with EMV-51, and Intellec® Series II or III Development Systems.

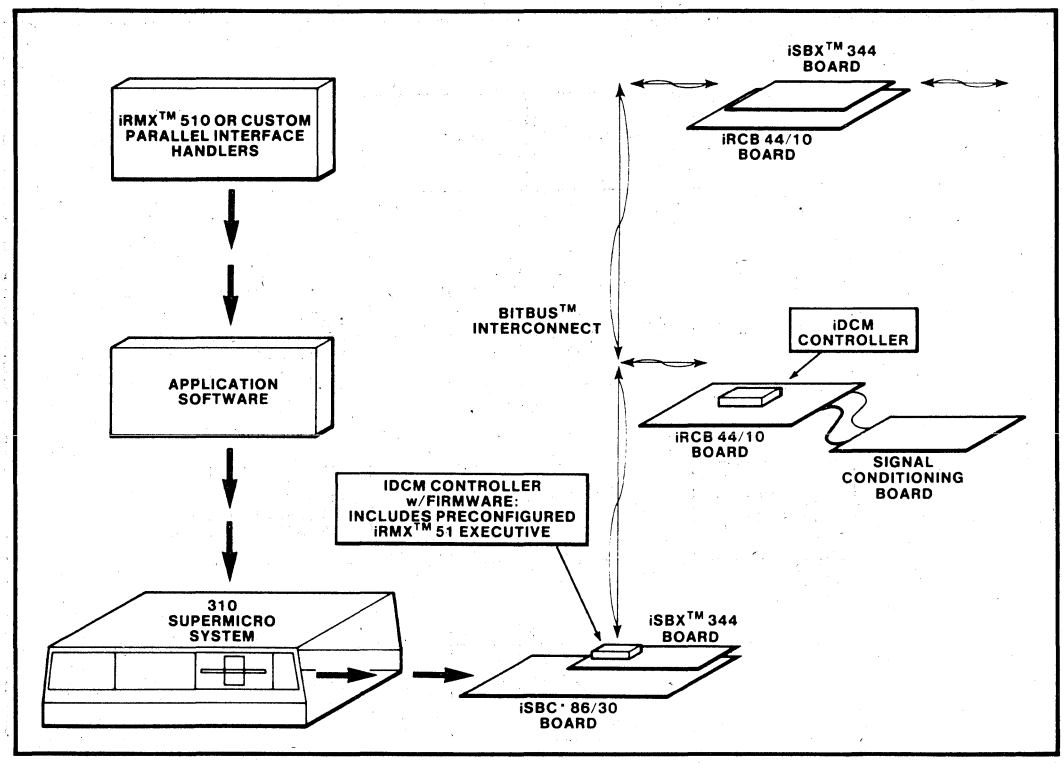


Figure 4. iDCM Operating Environment

SPECIFICATIONS

Supported Hardware

Microcontrollers

8051	80C51
8052	8044
8751	8744
8031	80C31
8032	8344

iDCM Product Line

iSBX 344 MULTIMODULE Board
iRCB 44/10 Remote Controller Board

Compatible Software

iRMX™ 510 iDCM Support Package

Development Tools

ICE™ 51 or ICE 44 Emulators
Intellec Series II or III Development System
iPDS System w/EMV-51
iRMX 510 iDCM Support Package
8051 Software Development Package

Reference Manual (Supplied)

146312-001 — Guide to Using the Distributed Control Modules

Ordering Information

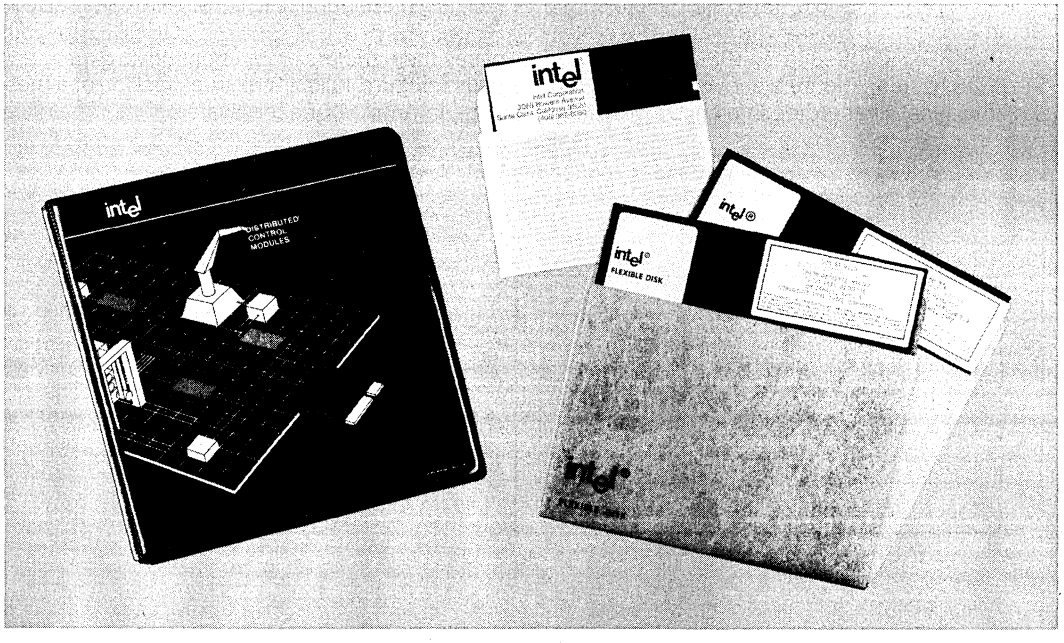
Part Number	Description
iRMX 51BY	Executive for 8051 Family of Microcontrollers with Reference Manual. A, B, and F Media Formats Supplied

iRMX™ 510
iDCM SUPPORT PACKAGE

- **Low cost remote communication/ control expansion for MULTIBUS® based systems**
- **Extends functionality of BITBUS™/iDCM systems**
- **Software development support for BITBUS™/iDCM products: iSBX™ 344 and iRCB 44/10 boards**
- **Simple software interface for iRMX™ 86, 286, 88, and iPDS™ ISIS operating system compatibility**

The iRMX™ 510 iDCM Support Package contains the necessary software tools to interface MULTIBUS®, and iPDS™ ISIS systems to BITBUS™ systems in both a development environment and during run-time. With other members of the Distributed Control Modules family, the iRMX 510 iDCM Support Package expands Intel's OEM Microcomputer Systems capabilities to include distributed real-time control.

The iRMX 510 Package software interface handlers and the iSBX™ 344 BITBUS Controller MULTI-MODULE™ board extend the capabilities of other microprocessors such as the 8086, 80186, or 80286 in iDCM, MULTIBUS, or iPDS systems. Support of iRMX 51 applications is provided via the iRMX 51 libraries incorporated in the iRMX 510 Support Package. Also, the Support Package completes the development environment for BITBUS/iDCM products : iSBX 344 and iRCB 44/10 boards. When used with an ICE-44 Emulator the iDCM controller is accurately simulated resulting in a highly effective product development effort.



ORDER NUMBER: 230972-001

MULTIBUS®, iPDS,™ and iDCM SYSTEM EXPANSION

The iRMX 510 Support Package provides the software interface between Intel's MULTIBUS and iPDS environment, and the BITBUS environment. With Intel's Distributed Control Modules hardware interface, the iSBX 344 MULTIMODULE board, this capability enables the user to expand the existing functionality of an iRMX-based SYSTEM 310, for example, to include control and monitoring of a material handling operation. Intel's Personal Development System (iPDS) can be used as a central supervisory station for data acquistion in a laboratory or for program development. The iRMX 510 iDCM Support Package provides a general purpose interface. For custom applications, users may wish to develop a custom interface.

OPERATING ENVIRONMENT

The iRMX 510 Support Package is supplied on diskettes formated for iRMX, Intellec® Series II or III and iPDS ISIS development systems. Application programs or tasks residing on an extension in the iDCM environment may use the iRMX 510 interface. (Application programs or tasks are written in iRMX 88, 86 or ISIS compatible code.) Some examples of extensions in an iDCM system are the iSBC 86/05, 88/25, 186/03 boards and the iPDS system. Figure 2 shows how the iRMX 510 interface is integrated into an iDCM system.

For iRMX 86, 88, or 286R-based systems, configuration of the iRMX 510 interface requires two steps: configuring the interface to the hardware and then the supporting executive. Hardware configuration requires creating a file of configuration parameters, compiling it, and linking the result with the application program. When using the iRMX 510 Package with the iPDS ISIS system, hardware configuration is not required.

ARCHITECTURE

The major functional blocks of the iRMX 510 Support Package are: iRMX 86, 286R, 88 and iPDS ISIS parallel interface handlers, iDCM Controller firmware files, and iRMX 51 include files.

Simple Parallel Interface Handlers

The iRMX 510 Support Package includes parallel interface handlers for systems using the iRMX 86 or 286R Operating System, the iRMX 88 Executive, or Intel's Personal Development System ISIS Operating System. These software handlers pass iRMX 51 messages to and from the iSBX 344 parallel interface (Byte FIFO). In iRMX 86, 286R or 88 — based systems , the interface executes as two tasks: one to transmit, the other to receive the message. In iPDS systems the interface is a procedural call: DCM TRANSMIT, DCM RECIEVE, or DCM STATUS CHECK. In both cases the handlers are straightforward and easy to use. Figure 1 illustrates transmission of a message in an iRMX-based system.

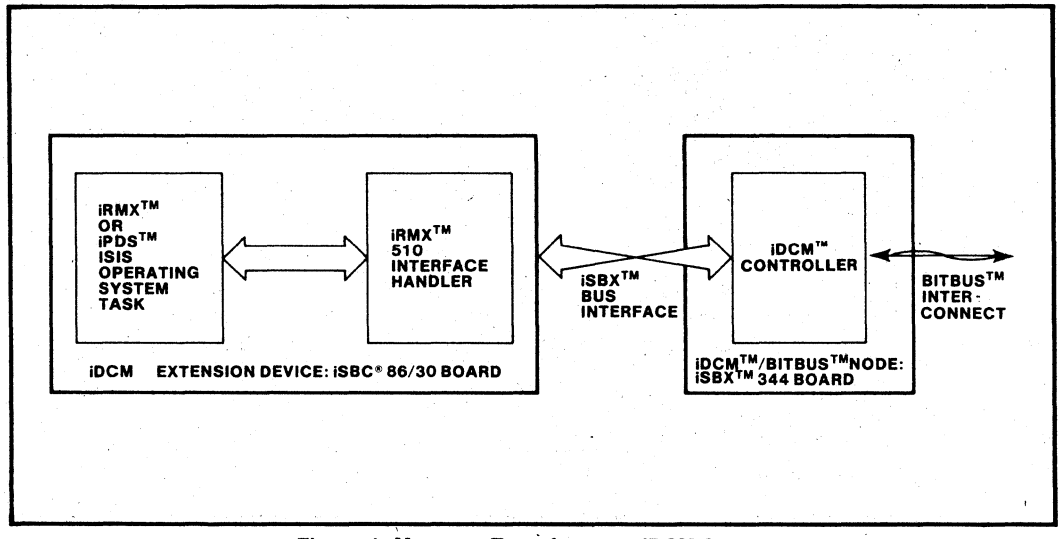


Figure 1. Message Transfer to an iDCM System

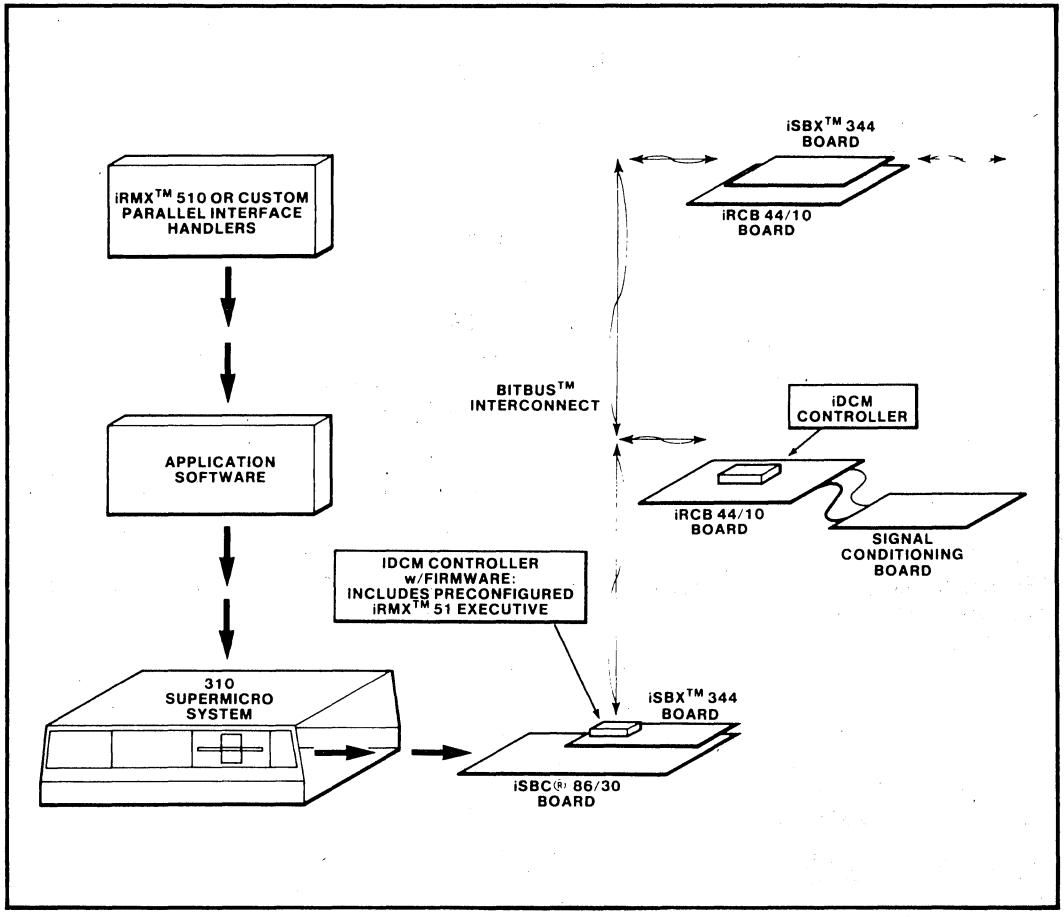


Figure 2. iDCM Operating Environment

The software handlers ease integration of other processors into an iDCM system and provide the tools to quickly expand a MULTIBUS system, or an iPDS ISIS system. Significant reduction in application system software development time results, with more effort concentrated on the overall application.

iDCM Controller Firmware

Also included in the iRMX 510 Support Package is the iDCM Controller firmware in loadable object files, iRMX 51 libraries, and iDCM Controller Include files. An Intellec Development System and ICE-44 Emulator can be used with the loadable object files to accurately simulate the iDCM Controller. This capability significantly decreases development effort by reducing trial and error production of application system software. The iRMX 51 Interface Library and iDCM Controller Include files allow development of user code for iDCM systems.

DEVELOPMENT ENVIRONMENT

The iRMX 510 Support Package completes the development environment for iDCM application system development when used with an Intellec Series II or III Development System and In-Circuit Emulator (ICE-44), or an iPDS system EMV-440 and the 8051 Software Development Package. As part of Intel's complete development environment for the 8051 family of microcontrollers, the iRMX 510 Support Package may also be used with an iPDS system and EMV-51 or an Intellec Series II or III Development System and an ICE-51 Emulator.

SPECIFICATIONS

Supported Hardware/Software for iDCM Systems

Operating System	Supported Extension*
iRMX 86 Release 5.0	iSBC 86/05, 86/14, 86/30, 186/03, 186/51, 188/48, 88/25, 88/45 boards
iRMX 88 Release 3.0	iSBC 86/05, 86/14, 86/30, 186/03, 186/51, 188/48, 88/25, 88/45 boards
iRMX 286R	iSBC 286/10 board
ISIS Release 1.0 (PDS)	iPDS System

*Each extension device uses an iSBX 344 BIT-BUS Controller MULTIMODULE Board

Supported Hardware — 8051 Microcontroller Family

8051	80C51
8052	8044
8751	8744
8031	80C31
8032	8344

Compatible Software
iRMX 86 Release 5.0
iRMX 286R
iRMX 88 Release 3.0
iPDS ISIS Release 1.0
iRMX 51 Release 1.0

Development Tools
ICE-51 or ICE-44 Emulators
iPDS System with EMV-51
Intellec Series II or III Development System
8051 Software Development Package

Reference Manual

146312-001 — Guide to Using the Distributed Control Modules (Supplied)

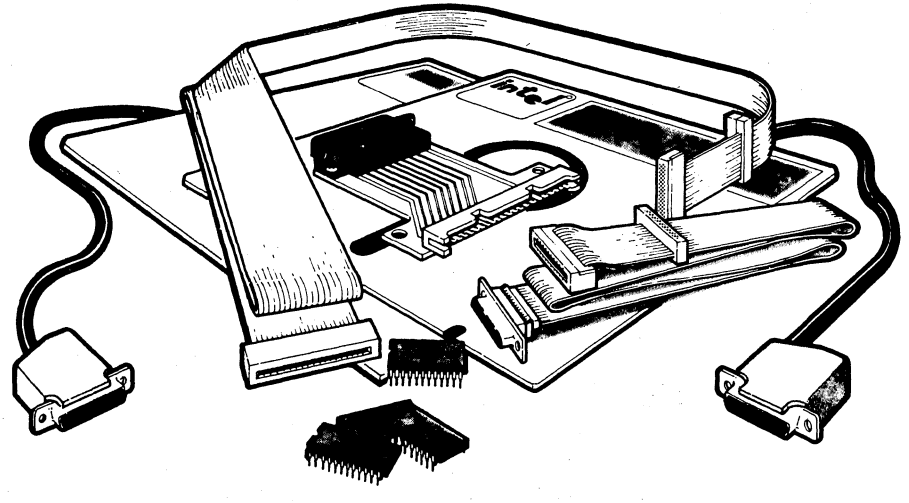
Ordering Information

Part Number	Description
iRMX 510BY	iDCM Support Package w/ Reference Manual A,B,E, and F Media Formats Supplied.

iSDM™ 86
SYSTEM DEBUG MONITOR

- **Supports target system debugging for iSBC® /iAPX 86, 88, 186 and 188-based applications**
- **Provides interactive debugging commands including single-step code execution and symbolic displays of results**
- **Supports 8087 Numeric Processor Extension (NPX) for high-speed math applications**
- **Allows building of custom commands through the Command Extension Interface (CEI)**
- **Supports application access to ISIS-II files**
- **Provides program load capability from an Intellec® Development System**
- **Contains configuration facilities which allow an applications bootstrap from iRMX™ 86 and 88 file compatible peripherals**
- **Modular to allow use from an Intellec Development System or from a stand-alone terminal**

The Intel iSDM™ 86 System Debug Monitor package contains the necessary hardware, software, cables, EPROMs and documentation required to interface, through a serial or parallel connection, an iSBC® 86/05, 86/12A, 86/14, 86/30, 88/25, 88/40, 88/45, 186/03, 186/51, 188/48, or iAPX 86, 88, 186 or 188 target system to an MDS 800, Series II, Series III, or Series IV Intellec® Microcomputer Development System for execution and interactive debugging of applications software on the target system. The Monitor can: load programs into the target system; execute the programs instruction by instruction or at full speed; set breakpoints; and examine/modify CPU registers, memory content, and other crucial environmental details. Additional custom commands can be built using the Command Extension Interface (CEI). The Monitor supports the OEM's choice of the iRMX™ 86 Operating System, the iRMX 88 Real-Time Multi-tasking Executive or a custom system for the target application system. OEM's may utilize any iRMX 86, 88 supported target system peripheral for a bootstrap of the application system or have full access to the ISIS-II files of the Intellect System.



October, 1983
ORDER NUMBER: 230882-002

FUNCTIONAL DESCRIPTION

Overview

The iSDM 86 Monitor extends the software development capabilities of the Intellec system so the user can effectively develop applications to ensure timely product availability.

The iSDM 86 package consists of four parts:

- The loader program
- The iSDM 86 Monitor
- The Command Extension Interface (CEI)
- The ISIS-II Interface

The user can use the ISDM 86 package to load programs into the target system from the development system, execute programs in an instruction-by-instruction manner, and add custom commands through the command extension interface. The user also has the option of using just the iSDM 86 Monitor and the CEI in a stand-alone application, without the use of an Intellec development system.

Powerful Debugging Commands

The iSDM 86 Monitor contains a powerful set of commands to support the debugging process. Some of the features included are: bootstrap of application software; selective execution of program modules based on breakpoints or single stepping requests; examination, modification and movement of memory contents; examination and modification of CPU registers, including NPX registers. All results are displayed in clearly understandable formats. Refer to Table 1 for a more detailed list of the iSDM 86 monitor commands.

Numeric Data Processor Support

Arithmetic applications utilizing the 8087 Numeric Processor Extension (NPX) are fully supported by the iSDM 86 Monitor. In addition to executing applications with the full NPX performance, users may examine and modify the NPX's registers using decimal and real number format.

This feature allows the user to feel confident that correct and meaningful numbers are entered for the application without having to encode and decode complex real, integer, and BCD hexadecimal formats.

Command Extension Interface (CEI)

The Command Extension Interface (CEI) allows the addition of custom commands to the iSDM 86 Monitor commands. The CEI consists of various procedures that can be used to generate custom commands. Up to three custom commands (or sets of commands) can be added

Table 1. Monitor Commands

Command	Function
B	**Bootstrap** application program from target systems peripheral device
C	**Compare** two memory blocks
D	**Display** contents of memory block
E*	**Exit** from loader program to ISIS-II Interface
F	**Find** specified constant in a memory block
G	**Execute** application program
I	**Input** and display data obtained from input port
L*	**Load** absolute Intellec® object file into target system memory
M	**Move** contents of memory block to another location
N	**Display and execute** single instruction
O	**Output** data to output port
P	**Print** values of literals
R*	**Load and execute** absolute Intellec® object file in target system memory
S	**Display and (optionally) modify** contents of memory
T*	**Transfer** block of memory to an Intellec® file
U,V,W	**User** defined custom commands extensions
X	**Examine and (optionally) modify** CPU and NPX registers

* Commands require an attached Series II/Series III.

to the monitor without programming new EPROMs or changing the monitor's source code.

ISIS-II Interface

The ISIS-II interface consists of libraries which contain interfaces to ISIS-II I/O calls. A program running on an iAPX 86, 88, 186 or 188-based system can use the ISIS-II interface and access the individual ISIS-II I/O calls. The interface allows the inclusion of these calls into the program; however, most of the calls require a Series II/ Series III system. Table 2 contains a summary of the major I/O calls and parameters.

Program Load Capability

The iSDM 86 loader allows the loading of iAPX 86, 88, 186 or 188-based programs into the target system. It executes on a Intellec Microcomputer Development System and communicates with the target system through a serial or a parallel load interface. If a Series II/Series III/Series IV system containing an Intel I/O expansion board is being used, the board can be used as a fast parallel load interface, freeing up the UPP port for application use.

Configuration Facility

The monitor contains a full set of configuration facilities which allow it to be carefully tailored to the requirements of the target system. Pre-configured EPROM-resident monitors are supplied by Intel for the iSBC 86/05, 86/12A, 86/14, 86/30, 88/45, 186/03, 186/51, and 188/48 boards. The monitor must be configured by the user for the iSBC 88/25, 88/40 boards and for other iAPX 86, 88, 186, 188 applications. iRMX 86 and iRMX 88 system users may use the configuration facilities to include the iAPX 86, 88 Bootstrap Loader (V5.0 or newer) in the monitor.

Variety of Connections Available

The physical interface between the Intellec Microcomputer Development System and the target system can be established in one of three ways. The systems can be connected via a serial link, a parallel link or a fast parallel link. The fast parallel link requires the use of an iSBC 108(A), 116(A), 517 or 519 I/O expansion board in the Intellec system and is only available for connections with the Series II/Series III/Series IV systems. The cabling arrangement is different depending upon the development system being used. Figure 1 displays the cable connections needed between an Intellec Series III system and a target system for a serial interface.

The iSDM 86 Monitor does not require the use of a development system. The monitor can be used by simply attaching a stand-alone terminal to the target system. Figure 1 also displays the cable connections needed for this arrangement.

Table 2. Routines for ISIS-II Services Available to Target System Applications

Routine	Target System Function
ATTRIB	Changes to ISIS-II file **attribute**
CI	Returns a character **input** from the **console**
CO	Transfers a character for **console output**
CLOSE	**Closes** an opened ISIS-II file
DELETE	**Deletes** the specified ISIS-II file
DQ$CFG	Returns information about monitor's communication link and type
ERROR	Displays an **error** message on the Intellec® console
EXIT	**Exits** to the target system monitor
LOAD	**Loads** target system memory with ISIS-II object code file
OPEN	**Opens** an ISIS-II file for access
READ	**Reads** up to 4096 bytes from an ISIS-II file to memory
RENAME	**Renames** an ISIS-II disk file
SEEK	**Seeks** to the specified ISIS-II file location
WRITE	**Writes** up to 4096 bytes from memory to an ISIS-II file

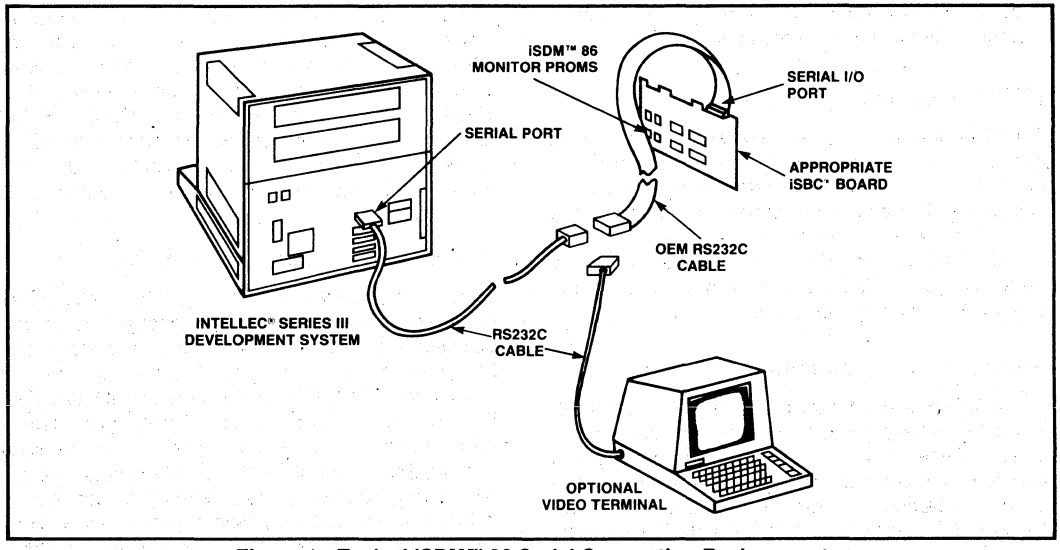


Figure 1. Typical iSDM™ 86 Serial Connection Environment

SPECIFICATIONS

Development System Environment

The Intellec Microcomputer Development System may be utilized for application program development and, if used, requires the following to support the iSDM 86 package:

- 48 Kbytes memory
- Double density or single density diskette subsystem
- ISIS-II Operating System and associated language translators

iAPX 86, 88, 186, 188 TARGET SYSTEM ENVIRONMENT

To support the iSDM 86 package, the target system must contain the following:

- 2K read-write memory beginning at location 0H
- 16K read-only memory beginning at location FC000H
- For Parallel link:
  — 8255A Programmable Peripheral Interface
- For Serial link:
  — 8251A USART or 8274 Multiprotocol Serial Controller, and 8253/4 or 80130 or iAPX 186/188 timer, or
  — 82530 Serial Communications Controller, including 82530 timer

Hardware

- Supported iSBC Microcomputers:

iSBC 86/05	Single Board Computer
iSBC 86/12A	Single Board Computer
iSBC 86/14	Single Board Computer
iSBC 86/30	Single Board Computer
iSBC 88/25	Single Board Computer
iSBC 88/40	Single Board Computer
iSBC 88/45	Single Board Computer
iSBC 186/03	Single Board Computer
iSBC 186/51	Single Board Computer
iSBC 188/48	Single Board Computer

- Supported iSBX MULTIMODULE™ Boards:

  iSBX 350 Parallel I/O MULTIMODULE Board
  iSBX 351 Serial I/O MULTIMODULE Board

iSDM™ 86 Package Contents

Cables:

1 — Parallel I/O Cable (upload/download)

2 — RS232 Cables

Adaptors:

1 — Parallel Status Adaptor

1 — Parallel Adaptor

I/O Drivers and Terminators:

4 — Pull-up Resistor Packs

4 — Pull-up/down Resistor Packs

4 — Line Driver Packs

Interface and Execution Software Diskettes:

1 — Single Density, ISIS Compatible

1 — Double Density, ISIS Compatible

System Monitor EPROMs:

Microcomputer	EPROM
iSBC® 86/05 iSBC® 86/12A iSBC® 86/14 iSBC® 86/30	Four 2732A EPROMs
iSBC® 88/45	Two 2764 EPROMs
iSBC® 186/03 iSBC® 186/51	Two 2764 EPROMs
iSBC® 188/48	Two 2764 EPROMs

Reference Manual (Supplied):

146165-001 — iSDM 86 System Debug Moniter Reference Manual

ORDERING INFORMATION

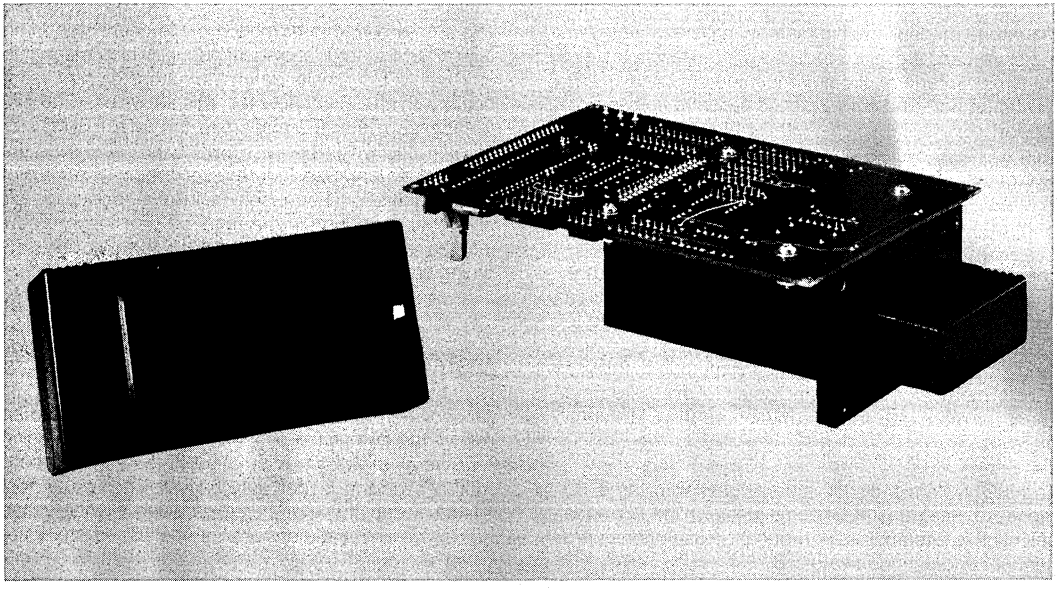
Part Number	Description
iSDM 86	Intellec to target system interface and target system monitor, suitable for use on iSBC 86, 88, 186, 188 computers, or other iAPX 86, 88, 186, 188 microcomputers. Package includes cables, EPROMs, software and operator manual. The iSDM 86 package includes SPR Service for 90 days after shipment. As with all Intel Software, purchase of any of these options requires execution of a standard Intel Master Software License.
iSDM 86 RO	Object Software
iSDM 86 BSR	Machine Readable Source

Memory Expansion Boards 7

iBCK 10-1 and iBCK 10-4
INTEL BUBBLE CASSETTE SYSTEM PRODUCTION KIT

- **iBCK 10-1 0-65°C**
  iBCK 10-4 10-50°C
  At 20% Average Duty Cycle
- **128 KBytes of Non-Volatile Bubble Memory Per Cassette**
- **Simple Interface: Complete iSBX™ Compatibility optional using iSBX™ iBC Interface MULTIMODULE™**
- **Performance:**
  —Average Access Time: 48 msec
  —Burst Data Rate: 12.5 KBytes/Sec
- **High Reliability Under Harsh Environments**
- **2 Cassette to Holder Keying Options**
- **Power Fail Data Protection**
- **Operates under Standard +5 and +12Vdc Power Supplies**
- **Bubble Detector Switching Saves Power**
- **Write Protect Mode**

The Intel iBCK 10-1 and iBCK 10-4 Bubble Memory Cassette Systems are fully assembled and tested removeable nonvolatile storage systems based on the Intel 7110A one megabit bubble memory and support components. Each iBCK 10-1 and iBCK 10-4 comes with one iBCH 110 Bubble Cassette Holder and one iBC 128-1 or iBC 128-4 bubble cassette. Extra bubble cassettes are available separately. The 128 KByte bubble cassettes are completely interchangeable from one 128 KByte bubble cassette holder to any other; the operating temperature range of the system is determined by the bubble cassette used. Cassette-to-holder keying is accomplished using grooves on the front bezel of the cassette holder which must match those on the bottom of the cassette to allow entry of the cassette. A sliding tab on the cassette places the bubble memory in the write protected mode.



JUNE 1984
ORDER NUMBER: 231084-001

The Intel Bubble Cassette System offers OEM systems designers a rugged, reliable high performance alternative to disk and tape drives. The iBC system will function reliably in the hot, dusty, humid environments where tapes and disks will not. And because of its solid state memory technology and simple packaging, it can operate through high levels of shock and vibration. The system has only one mechanical or moving part, the cassette-to-holder connector. This is an industry-standard AMP D-type pin connector finished with AMP's Duragold and rated at 10,000 insertions for the male half (mounted in the holder) and 5000 insertions for the female half (mounted in the cassette).

The iBCK 10-1 and iBCK 10-4 bubble cassette systems may be designed into Intel SBC-based systems using the optional iSBX 258 iBC Interface Multimodule. The iBC system may also be designed into any user designed microprocessor board. The bubble memory support circuitry and ribbon cable connector on the bubble cassette holder board provide the user with a simple interface to the bubble memory.

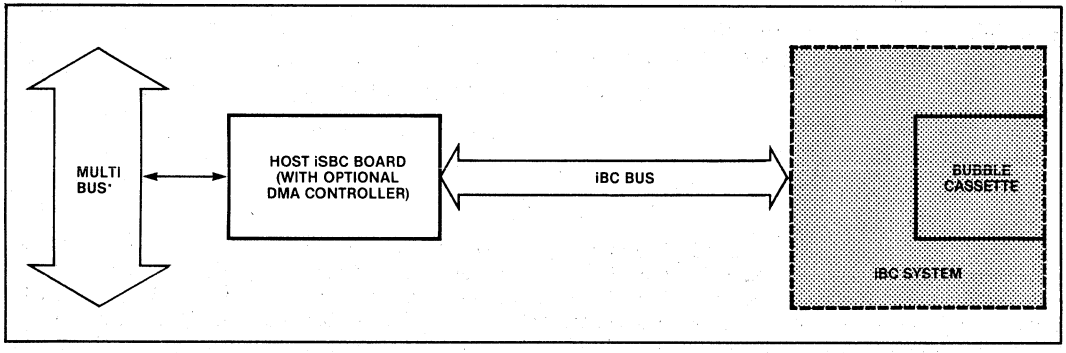


Figure 1. Block Diagram of MULTIBUS® iBC Configuration

FUNCTIONAL DESCRIPTION

The iBCK 10-1 and iBCK 10-4 bubble memory cassette systems are completely assembled and tested non-volatile memory systems. They consist of an Intel 7110A one megabit bubble memory device mounted in a plastic cassette and all of the 7110A's support circuitry plus interface circuitry mounted on a printed circuit board screwed onto a plastic cassette holder. The support circuitry includes the Intel 7220 Bubble Memory Controller (BMC) through which the host processor communicates with the bubble memory. See Figure 1 for a system and bubble memory block diagram.

The BMC provides a convenient 8 bit bi-directional data bus which requires only two port or I/O addresses. One port is used to transfer data while the other is used to send commands or view operational status. A set of sixteen commands are available to initiate and monitor a bubble memory data transfer (refer to the 7220 data sheet for more detailed information on the BMC commands).

The iBC system provides the designer with three I/O modes of data transfer for complete flexibility. I/O mode selection is accomplished through the use of on-board jumpers and user software:

1. Polled
2. Interrupt-driven mode
3. Direct Memory Access (DMA) mode

DMA mode requires the use of a DMA controller on the host board.

The iBC system can be ribbon cable connected to an iSBX 258 iBC Interface Multimodule which plugs into any Intel iSBC Single Board Computer or processor board with an iSBX connector. This allows easy iSBX interface from the iBC and frees the MULTIBUS® for other traffic while the host iSBC board accesses the bubble memory. See Figure 2 for a block diagram of this configuration. For ribbon cable lengths of up to 18″, no signal drive circuitry is required. For longer cable lengths, sufficient drive circuitry to maintain iBC bus timing specifications must be used.

MOUNTING TECHNIQUE

As shown in Figure 3, the iBC holder screws into an opening in an outside panel of the host system using four screws. Additionally, the printed circuit board mounted on the holder may be screwed to the host system using the four holes on each corner of the board. Pertinent dimensions of the iBC system are given in Figure 3.

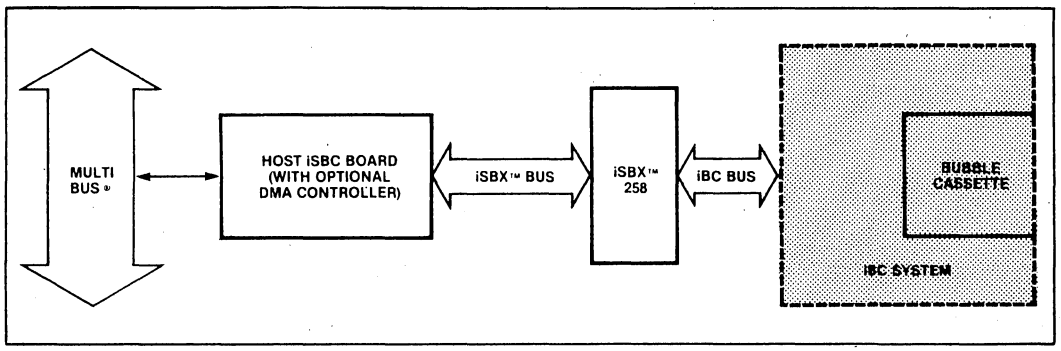


Figure 2. MULTIMODULE™ Mounting

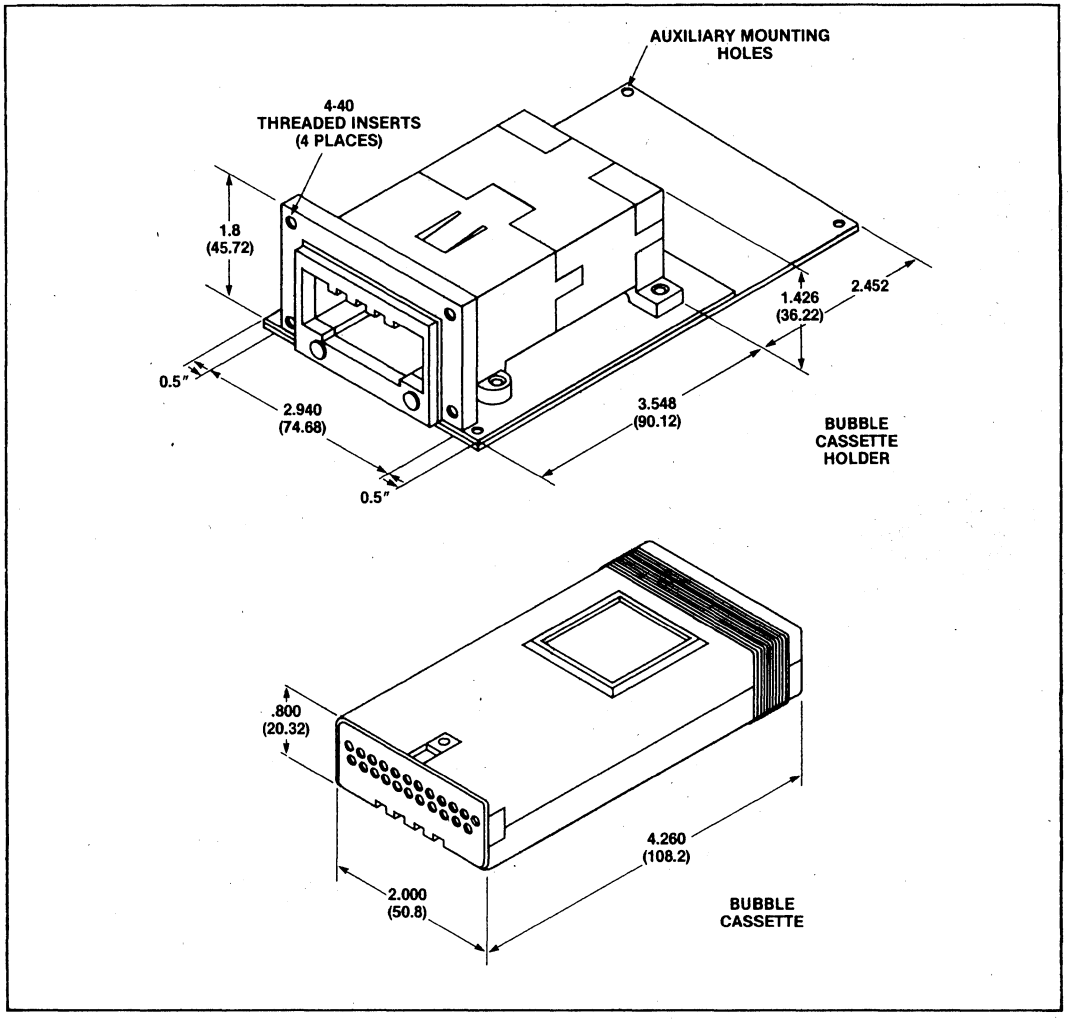


Figure 3. Dimensions and Mounting

SPECIFICATIONS

Storage Capacity

— 128K Bytes per cassette
— 2048 Pages
— Page Length:
  64 Bytes with ECC
  68 Bytes without ECC

Physical Characteristics

Width: 10.16 cm (4.00 in.) max.
Length: 15.24 cm (6.00 in.) max.
Height: 4.57 cm (1.80 in.) max.
Weight: 139 grams (4.9 oz.) holder only
  218 grams (7.2 oz.) cassette only

Interface Requirements

— TTL compatible
— 40 pin ribbon cable

Shock/Vibration

System Vibration:
Mil Std 810C, Method 514.2,
Curve V, 5-200-5 Hz on 3 axes, 1.5G's
Constant Acceleration

System Shock:
Mil Std 810C, Method 516, Procedure 1, 30G's,
½ sine, 11 msec.

Operational Modes

Polled, Interrupt Driven, or DMA (with Host DMA Controller)

Environment

	Average Duty Cycle	Operating Temperature
iBCK 10-1	10%	0-65°C
	30%	0-63°C
	60%	0-60°C
iBCK 10-4	10%	10-52°C
	30%	10-48°C
	60%	10-40°C

Temperatures quoted are ambient with 100 lfm air flow.

Relative Humidity

0% to 95% without condensation

Non-Volatile Storage Temperature:

iBCK 10-4: −20 to +75°C
iBCK 10-1: −40 to +90°C

Electrical Requirements

D.C. power supplied through external connector.

Voltage	Tolerance	Power Off/Power Fail Decay Rate	Max. Active Current	Typical Standby Current
+12 Volts	±5%	less than 1.10 volts/msec	400 mA	35 mA
+5 Volts	±5%	less than 0.45 volts/msec	570 mA	200 mA

Additional Documentation

Intel Bubble Cassette System Users' Manual
(Order #122278-001)

iBCK 12
INTEL BUBBLE CASSETTE SYSTEM PROTOTYPING KIT

- **Complete iBC System Plus iSBX™ 258 iBC Interface MULTIMODULE™ and Ribbon Cable**
- **128 KBytes of Removeable, Reliable, Non-Volatile Bubble Memory storage per Cassette**
- **Performance:**
  —Average Access Time: 48 msec
  —Burst Data Rate: 12.5 KBytes/Sec
- **Fastest, easiest way to design with the iBC system; ready to use as a peripheral on the iSBX bus.**

The Intel iBCK 12 Intel Bubble Cassette (iBC) System Prototyping Kit contains all the necessary items to begin development with the iBC system. This is a 128 KByte Intel Bubble Cassette (iBC 128), Intel Bubble Cassette Holder (iBCH 110), an iSBX Interface MULTIMODULE for the iBC (iSBX 258), and an 18" ribbon cable to connect them. The kit allows systems designers to prototype the iBC system in their systems with a minimum of time and effort.

The iBC system is described by its data sheet (iBCK 10-1 and iBCK 10-4) in more detail, as is the iSBX 258 by its data sheet.

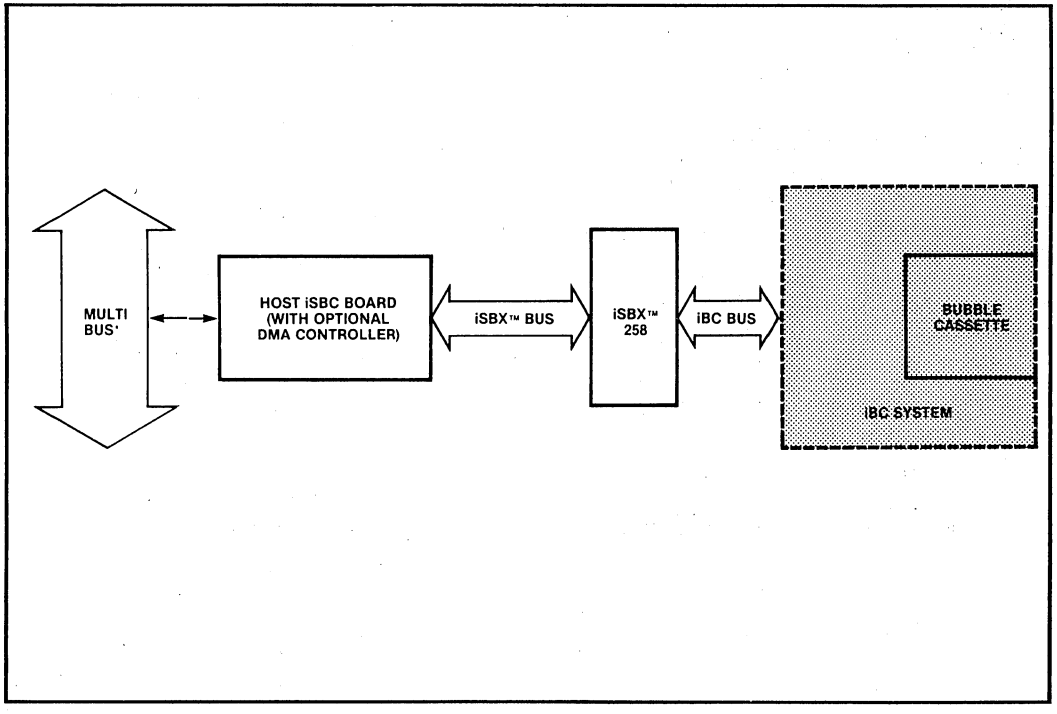


Figure 1. Block Diagram

JUNE 1984
ORDER NUMBER: 231086-001

FUNCTIONAL DESCRIPTION

The iBCK 12 is a complete solution for prototyping with the iBC System. Steps to build up a prototyping system using the iBCK 12 are:

1. Plug iSBX 258 single-wide MULITMODULE onto host board with iSBX connector;
2. Connect the iBC holder board to the iSBX 258 using the the ribbon cable included in the kit;
3. Connect +5V and +12V to the iBC holder board using the power connectors included in the kit;
4. Insert the Intel Bubble Cassette in the iBC holder.

See Figure 1 for a block diagram of the configuration.

SPECIFICATIONS

Storage Capacity

— 128K Bytes per cassette
— 2048 Pages
— Page Length:
64 Bytes with ECC

Performance

Maximum Data Rate: 100K bits/sec
Average Access Time: 48 ms
Average Transfer Rate: 68K bits/sec

Kit Contents

iBCH 110 — Intel Bubble Cassette Holder
iBC 128 — Intel Bubble Cassette (128KBytes)
iSBX 258 — iBC Interface MULTIMODULE™
18″ Ribbon Cable
Power Connector
Intel Bubble Cassette System User's Manual
(Order #122278-001)

Interface Requirements

— TTL compatible
— iSBX 258 male connector plugs into 36-pin or 44-pin host female connector

Operational Modes

Polled, Interrupt Driven, or DMA (with Host DMA Controller)

Temperature

+20 to +30° C operating @ 30% average duty cycle
-20 to + 75° C Non-Volatile Storage

Electrical Requirements

iBC System:
D.C. power supplied through external connector.

Voltage	Tolerance	Power Off/ Power Fail Decay Rate	Max. Active Current	Typical Standby Current
+12 Volts	±5%	less than 1.10 volts/msec	400 mA	35 mA
+5 Volts	±5%	less than 0.45 volts/msec	570 mA	200 mA

iSBX 258:
D.C. power, supplied through SBX connector.
+5V ±5%, 285 mA (max).

Additional Documentation

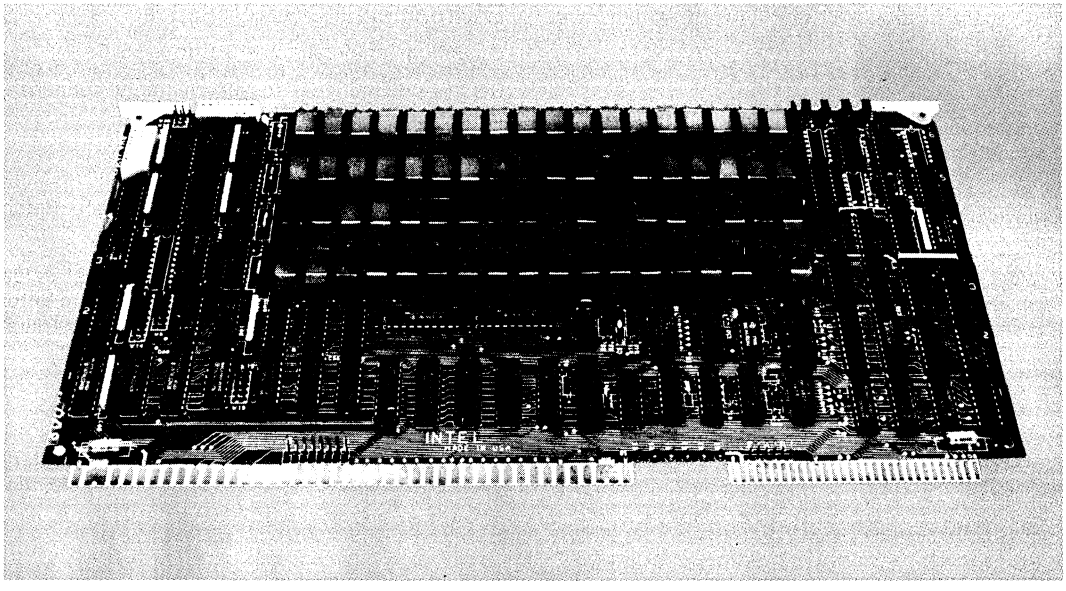
Intel Bubble Cassette System Users' Manual

iSBC® 012B
RAM MEMORY BOARDS

- **iSBC®86, iSBC®88 and iSBC®80 board RAM expansion through direct MULTIBUS® interface**
- **512K of read/write memory**
- **On-board parity generator/checker and error status register**
- **Requires a single +5 volt power supply**
- **Assignable anywhere within a 16 megabyte address space**
- **Jumper selectable base address on any 16K byte boundary**
- **Auxiliary power bus and memory protect control logic for battery backup RAM requirements**

The iSBC 012B RAM memory board is a member of Intel's complete line of iSBC memory and I/O expansion boards. The board interfaces directly to any iSBC 86, iSBC 88 or iSBC 80 Single Board Computer via the MULTIBUS interface to expand system RAM capacity. The iSBC 012B board contains 512K bytes of read/write memory implemented using dynamic RAM components. An on-board dynamic RAM controller refreshes a portion of these components every 16 microseconds. Each refresh cycle utilizes memory for 550 nanoseconds (maximum).

The iSBC 012B board generates byte oriented parity during all write operations and performs parity checking during all read operations. When a parity error is detected, the board can generate an interrupt on the MULTIBUS interface. In addition, the row and bank of the RAM array containing the error are stored in a Parity Flag Register. This register is accessible as a MULTIBUS I/O port. An on-board LED also provides a visual indication that a parity error has occurred.



July, 1981
143876

SPECIFICATIONS

Word Size

8 bits and 16 bits

Memory Size

524,288 bytes (iSBC 012B)

Access Time

330 nsec (worst case)
300 nsec (typical)

Cycle Times (Worst Case)

Read — 500 ns max.
Write — 500 ns max.
Refresh — 550 ns max.

Interface

All address, data and command signals are TTL compatible.

Address Selection

Memory — Base address is jumper selectable on any 16K byte boundary in a 16 megabyte address space. On-board RAM cannot cross a 4 megabyte address boundary.

Parity Flag Register — The I/O address of the Parity Flag Register is jumper selectable to be between 00H to 0FH or 40H to 4FH.

Connector

Edge connector — 86 pin double-sided PC edge connector with 0.156 in. contact centers.

Mating connector — Viking 3KH43/9AMK12 or equivalent.

Auxiliary Power

An auxiliary power bus is provided to allow separate power to RAM array for systems requiring battery backup of read/write memory. Selection of this auxiliary RAM power bus is made via jumpers on the board.

Memory Protect

An active-low TTL compatible memory protect signal is brought out on the auxiliary connector which, when asserted, disables read/write access to RAM on the board. This input is provided for the protection of RAM contents during system power-down sequences.

Physical Characteristics

Width — 12.00 in. (30.48 cm)
Height — 6.75 in. (17.15 cm)
Depth — 0.50 in. (1.27 cm)
Weight — 14 oz. (397 gm)

Electrical Characteristics

D.C. POWER REQUIREMENTS

All configurations require only +5 volts ±5%.

Normal System Operation (max.)
4.8A (worst case)
3.46A (typical)

Auxiliary Power No RAM Access (max.)
1.35A (worst case)
0.88A (typical)

Environmental Characteristics

Operating Temperature — 0°C to +55°C

Relative Humidity — to 90% (without condensation)

Reference Manual

143865-001 — iSBC 056B/012B Hardware Reference Manual (not supplied)

Manuals may be ordered from any Intel sales representative, distributor office, or from Intel Literature Dept., 3065 Bowers Avenue, Santa Clara, California 95051.

ORDERING INFORMATION

Part Number Description

SBC 012B	512K-Byte RAM Board with Parity

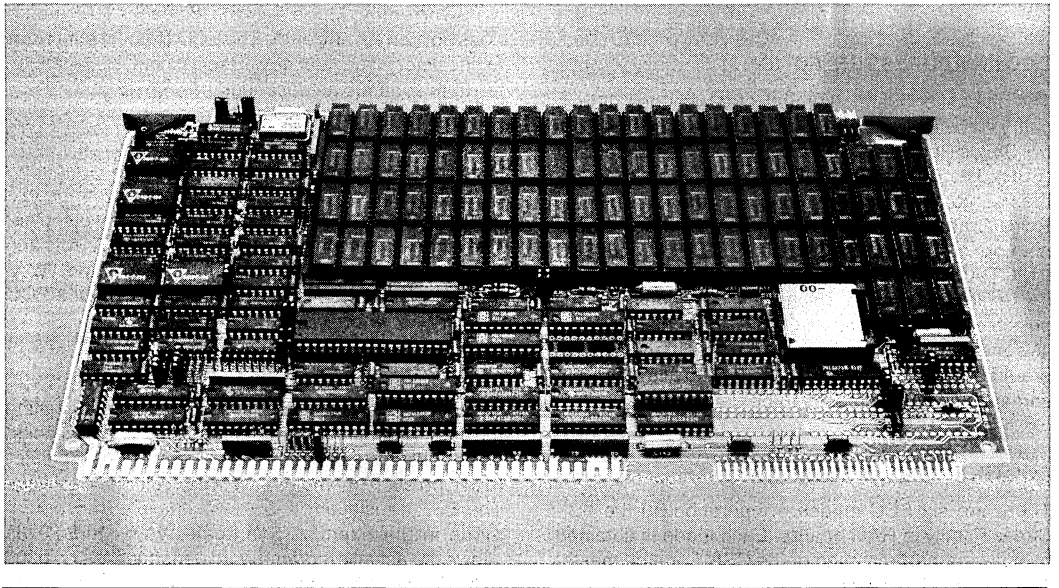
iSBC® 028C, 056C AND 012C ECC RAM BOARDS

- **iSBC® 86, iSBC® 88 RAM expansion through direct, IEEE P796, MULTIBUS® interface**
- **128K, 256K, or 512K bytes of read/write memory**
- **Single bit error correction and double bit error detection via Intel® 8206 ECC device**
- **Control status register supports multiple ECC operating modes**
- **Error status register provides error logging by host CPU board**
- **Base address selectable on 16K byte boundaries**
- **Supports 8 or 16-bit transfer and 24-bit addressing**
- **Auxiliary power bus and memory protect logic for battery back-up RAM requirements**

The iSBC® 028C, iSBC 056C and iSBC 012C RAM boards are members of Intel's complete line of iSBC memory and I/O Expansion boards. Each board interfaces directly to any iSBC 88 or iSBC 86 Single Board Computer via the IEEE P796 MULTIBUS® interface to expand system RAM capacity. The iSBC 028C, iSBC 056C and iSBC 012C boards contain 128K, 256K or 512K bytes of read/write memory implemented using dynamic RAM components.

Single bit error correction and double bit error detection are provided on the iSBC 028C, iSBC 056C and iSBC 012C boards via the Intel 8206 Error Checking and Correction (ECC) device. Due to the on-board ECC features of the board they are ideally suited in applications where integrity of the stored data is critical, such as financial transactions, process control and medical equipment applications.

Refresh control of the RAM array is handled on-board by the RAM Array Control Logic. Therefore, no external refresh commands are necessary.



FUNCTIONAL DESCRIPTION

General

The iSBC 028C, 056C, and 012C RAM boards are physically and electrically compatible with the MULTIBUS interface standard, IEEE P796, as outlined in the Intel MULTIBUS specification. The capacity of each RAM board in this series is determined by the number of RAM devices on-board.

System Memory Size

Maximum system memory size with this series of boards is 16 megabytes. On-board jumpers assign the board to one of four 4 megabyte pages. Each page is partitioned into 256 blocks of 16K bytes each. The smallest partition on any board in this series is 16K bytes. Jumpers assign the base address (lowest 16K block) within the selected 4 megabyte page.

Error Checking and Correcting (ECC)

Error Checking and Correction is accomplished with the Intel 8206 Error Checking and Correction device. This ECC component in conjunction with the ECC check bit RAM array provides error detection and correction of single bit errors and detection only of double bit and most multiple bit errors. The ECC circuitry can be programmed to various modes to provide full diagnostic testing of both the storage and check bit RAM arrays.

ECC I/O ADDRESS SELECTION

The processor board communicates with the ECC circuitry via a single I/O port. This port is used for the Control Status Register (CSR) and the Error Status Register (ESR). The Control Status Register is programmed by the user to determine the mode of operation while the Error Status Register provides information about memory errors. The iSBC 028C, iSBC 256C and iSBC 012C RAM boards are shipped with a Programmed Array Logic (PAL) device which allows selecting one of 9 possible addresses for the I/O port. The actual selection is done by jumper configuration. Additional unprogrammed locations are left in the PAL to allow application specific I/O addresses to be defined.

CONTROL STATUS REGISTER

There are six ECC modes of operation on the "C" Series Family of RAM boards. Each mode is obtained by software programming of the CSR from the master iSBC board. The six modes are:

a. Interrupt on any error mode
b. Interrupt on non-correctable error only mode
c. Correcting mode
d. Non-correcting mode
e. Diagnostic mode
f. Examine syndrome word mode

Modes (a) and (b) can be used in conjunction with (c) and (d). The six modes are described below.

Interrupt on Any Error Mode — In this mode the RAM board will interrupt the iSBC processor only when any error (single or multiple bit) is detected by the ECC circuitry.

Interrupt on Non-Correctable Error Mode — In this mode the RAM board will interrupt the iSBC processor only when a non-correctable (multiple bit) error is detected by the ECC circuitry. A multiple bit error is not correctable by the ECC circuitry.

Correcting Mode — In this mode the RAM board corrects any correctable error (single-bit error). Errors which are not correctable are not modified. Interrupts are generated depending on the interrupt mode selected.

Non-Correcting Mode — In this mode the RAM board does not correct any error. The ECC circuitry continues to check for errors, but no corrective action is taken. Interrupts continue as described previously.

Diagnostic Mode — This mode is used for testing the on-board ECC circuitry. In this mode the write enable strobe to the ECC RAM array is continuously disabled, The diagnostic mode can be used to simulate errors and in conjunction with the "Examine Syndrome Word Mode" examine the check bits generated by the ECC circuitry.

Examine Syndrome Word Mode — This mode, in conjunction with the "Diagnostic Mode", is used for testing the ECC memory. In this mode, the syndrome bits/ check bits are clocked into the Error Status Register (ESR) on every memory read/write cycle, respectively. The ESR translation PROM switches to a transparent mode in the Examine Syndrome Word Mode. This allows the actual syndrome word generated by the 8206 ECC device to be examined.

ERROR STATUS REGISTER

This 8-bit register contains information about memory errors. The ESR reflects the latest error occurance. Table 1 shows the status register format. Bits 5 & 6 show the failing row while bits 0 through 4 indicate which bit (of the 16-bit data word or the 6-bit ECC syndrome word) is in error. Bit 7 is always high.

Bit 6	Bit 5	Meaning	
0	0	Error in row	0
0	1		1
1	0		2
1	1		3

Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Meaning	
0	0	0	0	0	Error in data bit	0
0	0	0	0	1		1
0	0	0	1	0		2
0	0	0	1	1		3
0	0	1	0	0		4
0	0	1	0	1		5
0	0	1	1	0		6
0	0	1	1	1		7
0	1	0	0	0		8
0	1	0	0	1		9
0	1	0	1	0		10
0	1	0	1	1		11
0	1	1	0	0		12
0	1	1	0	1		13
0	1	1	1	0		14
0	1	1	1	1		15
1	0	0	0	0	Error in check bit	0
1	0	0	0	0		1
1	0	0	1	0		2
1	0	0	1	1		3
1	0	1	0	0		4
1	0	1	0	1		5
1	1	1	1	0	No Error	
1	1	1	1	1	Non-correctable (multiple-bit error)	

NOTE: Bit 7 is always high

Table 1.

Battery Back-up/Memory Protect

An auxiliary power bus is provided to allow separate power to the RAM array for systems requiring backup of read/write memory. An active low TTL compatible memory protect signal is brought out on the auxiliary bus connector which, when asserted, disables read/write access to the RAM board. This input is provided for the protection of RAM contents during system power-down sequences.

SPECIFICATIONS

Word Size Supported

8 or 16-bits

Memory Size

131,072 Bytes (iSBC 028C)

262,144 Bytes (iSBC 056C)

524,288 Bytes (iSBC 012C)

Access Times (All Densities)

Read/Full Write — 350 ns (max)

Write Byte — 530 ns (max)

Cycle Times (All Densities)

Read/Full Write — 460 ns (max)

Write Byte — 885 ns (max)

NOTE: If an error is detected, read access time and cycle times are extended by 255 ns.

Refresh Times

Refresh Cycle Time — 15.6 μs

Refresh Delay Time — 760 ns

Memory Partitioning

Maximum System RAM size is 16M Bytes

PAGE ADDRESS (4M BYTES)

1 of 4 megabyte pages as follows: 0-4 megabytes; 4-8 megabytes; 8-12 megabytes; 12-16 megabytes

BLOCK ADDRESS (16K BYTES)

iSBC 028C RAM board — 8 contiguous 16K Byte Blocks (128K Bytes)

iSBC 056C RAM board — 16 contiguous 16K Byte Blocks (256K Bytes)

iSBC 012C RAM board — 32 contiguous 16K Byte Blocks (512K Bytes)

NOTE: Blocks cannot cross 4M Byte Boundary.

BASE ADDRESS

Any 16K Byte Boundary

Power Requirements

Voltage — 5VDC ± 5%

Current — iSBC 028C 6.5A max; iSBC 056C 6.6A max; iSBC 012C 6.8A max

Standby — iSBC 028C 2.2A max (battery backup); iSBC 056C 2.3A max; iSBC 012C 2.5A max

Environmental Requirements

Operating Temperature — 0°C to 55°C

Operating Humidity — To 90% without condensation

Physical Dimensions

Width — 12 inches (30.48 cm)

Height — 6.75 inches (17.15 cm)

Thickness — 0.50 inches (1.27 cm)

Weight — iSBC 028C 16.7 ounces (4699 gm); iSBC 056C 19.0 ounces (5329 gm); iSBC 012C 23.5 ounces (6589 gm)

Reference Manuals

145183-001 — iSBC 028C/iSBC 056C/iSBC 012C Hardware Reference Manual

Manuals may be ordered from any Intel Sales Representative, Distributor Office, or from the Intel Literature Department, 3065 Bowers Avenue, Santa Clara, CA 95051.

ORDERING INFORMATION

Part Number	Description
SBC 012C	512K Byte RAM board with ECC
SBC 056C	256K Byte RAM board with ECC
SBC 028C	128K Byte RAM board with ECC

iSBC® 028CX, 056CX, 012CX, 010CX AND 020CX iLBX™ RAM BOARDS

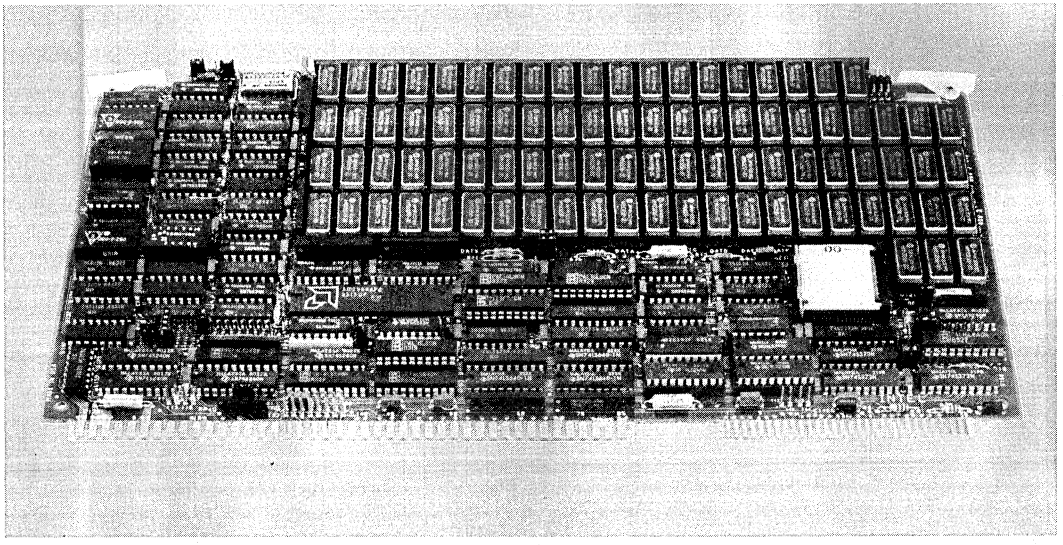
- **Dual port capability via MULTIBUS® and iLBX™ Interfaces**
- **Single bit error correction and double bit error detection utilizing Intel 8206 ECC device**
- **128K byte, 256K byte, 512K byte, 1024K byte, and 2048K byte versions available**
- **Control status register supports multiple ECC operating modes**
- **Error status register provides error logging by host CPU board**
- **16 megabyte addressing capability**
- **Supports 8- or 16-bit data transfer and 24-bit addressing**
- **Auxilliary power bus and memory protect logic for battery back-up RAM requirements**

The iSBC® 028CX, iSBC 056CX, iSBC 012CX, iSBC 010CX, and iSBC 020CX RAM memory boards are members of Intel's complete line of iSBC memory and I/O expansion boards. Each board interfaces directly to any iSBC 80, iSBC 86, iSBC 186 and iSBC 286 Single Board Computers. The dual port feature of the CX series of RAM-boards allows access to the memory of both the MULTIBUS® and iLBX™ bus interfaces.

In addition to the dual port features the "CX" series of RAM-boards provide Error Checking and Corrections Circuitry (ECC) which can detect and correct single bit errors and detect, but not correct, double and most multiple bit errors.

The iSBC 028CX, iSBC 056CX, and iSBC 012CX boards contain 128K, 256K, and 512K bytes of read/write memory using 64K dynamic RAM components. The iSBC 010 CX and iSBC 020 CX boards contain 1024K and 2048K bytes of read/write memory using 256K dynamic RAM components.

Due to the iLBX dual port capability and on-board ECC features of the boards they are ideally suited in applications where memory performance and integrity is critical, such as financial transactions, process control and medical equipment applications.



February, 1984
Order Number: 231023-001

FUNCTIONAL DESCRIPTION

General

The iSBC 028CX, 056CX, 012CX, 010CX and 020CX RAM boards are physically and electrically compatible with the MULTIBUS interface standard, IEEE-796, as outlined in the Intel MULTIBUS specification. In addition the CX series of RAM-boards are physically and electrically compatible with the iLBX bus (Local Bus Extension) interface as outlined in the Intel iLBX Specification (see Figure 1).

Dual Port Capabilities

The "CX" series of RAM-boards can be accessed by either the MULTIBUS interface or the iLBX interface (see Figure 2). Intel's iLBX interface is an unarbitrated bus architecture which allows direct transfer of data between the CPU and the memory boards without ac-

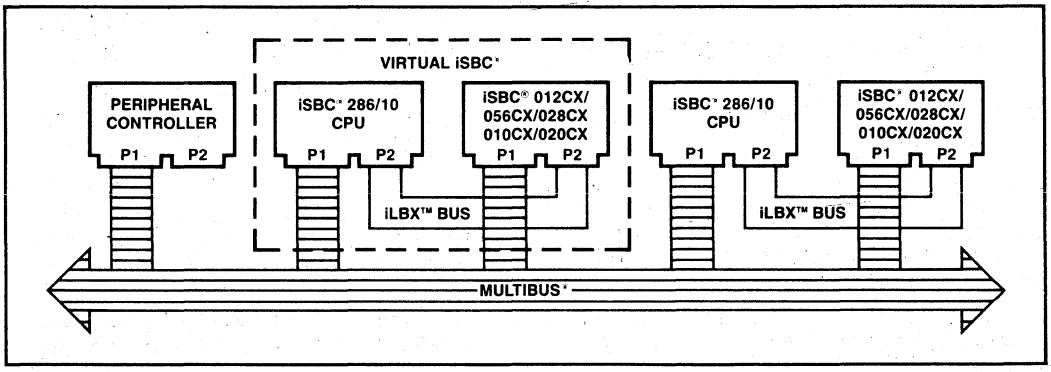


Figure 1. Typical iLBX™ System Configuration

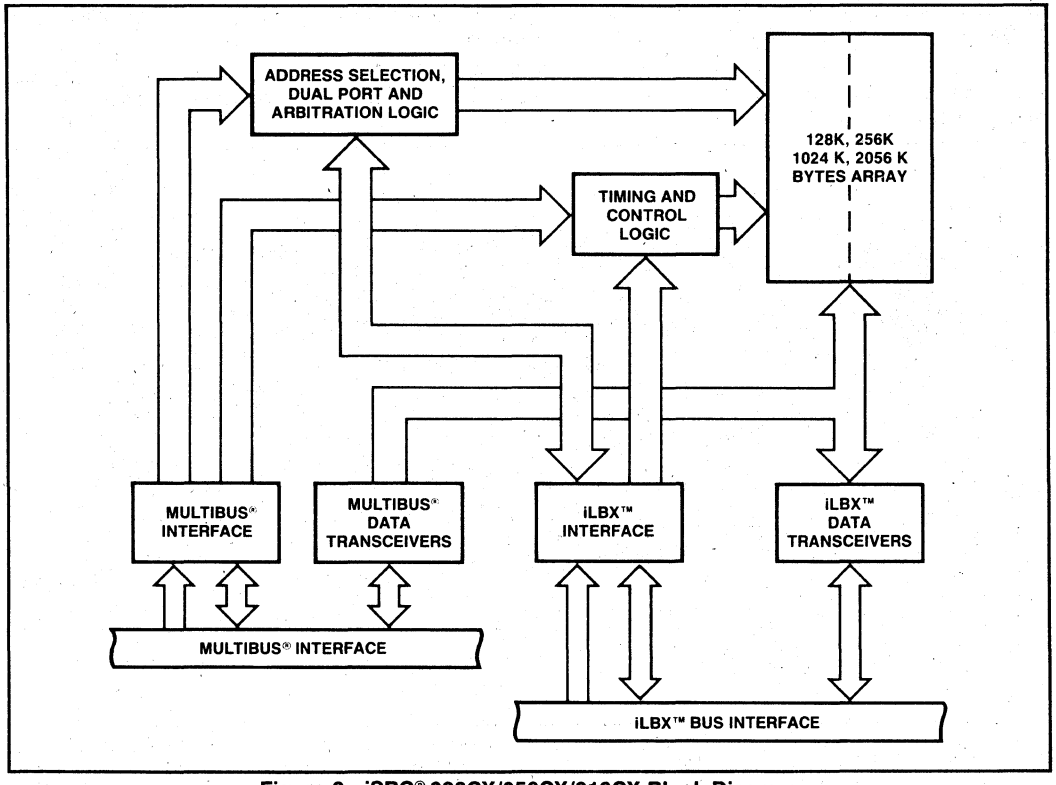


Figure 2. iSBC® 028CX/056CX/012CX Block Diagram

cessing the MULTIBUS bus. Due to the unarbitrated nature of the iLBX interface significant improvements in memory access times result, typically a 2-6 Wait State improvement over MULTIBUS memory access.

System Memory Size

Maximum system memory size with this series of boards is 16 megabytes. Memory partitioning is independent for the MULTIBUS interface and the iLBX interface.

For MULTIBUS operations, on-board jumpers assign the board to one of four 4-megabyte pages. Each page is partitioned into 256 blocks of 16K bytes each. The smallest partition on any board in this series is 8K bytes. Jumpers assign the base address (lowest 16K block) within the selected 4-megabyte page.

The iLBX bus memory partitioning differs from the MULTIBUS bus partitioning in that the iLBX bus address space consists of 256 contiguous blocks of 64K bytes totaling 16 megabytes. As with the MULTIBUS bus partitioning, the base addresses are set with on-board jumpers.

Error Checking and Correcting (ECC)

Error checking and correction is accomplished with the Intel 8206 Error Checking and Correcting device. This ECC component, in conjunction with the ECC check bit RAM array, provides error detection and correction of single bit errors and detection only of double bit and most multiple bit errors. The ECC circuitry can be programmed via the Control Status Register (CSR) to various modes while error logging is supported by the Error Status Register (ESR). Both CSR and ESR communicate with the master CPU board through a single I/O port.

ECC I/O Address Selection

The processor board communicates with the ECC circuitry via a single I/O port. This port is used for the Control Status Register (CSR) and the Error Status Register (ESR). The CSR is programmed by the user to determine the mode of operation while the ESR provides information about memory errors.

The iSBC 028CX, iSBC 056CX, iSBC 012CX, iSBC 010CX, and iSBC 020CX RAM boards are shipped with a Programmed Array Logic (PAL) device which allows selecting one of 9 possible addresses for the I/O port. The actual selection is done by jumper configuration. Additional unprogrammed locations are left in the PAL to allow application specific I/O addresses to be defined.

CONTROL STATUS REGISTER

There are six ECC modes of operation in the "CX" family of RAM boards. Each mode is obtained by software programming of the CSR from the master iSBC board. The six modes are:

a. Interrupt on any error mode
b. Interrupt on non-correctable error only mode
c. Correcting mode
d. Non-correcting mode
e. Diagnostic mode
f. Examine syndrome word mode

Modes (a) and (b) can be used in conjunction with (c) and (d). The six modes are described below.

Interrupt on Any Error Mode — In this mode the RAM board will interrupt the iSBC processor board when any error (single bit or multiple bit) is detected by the ECC circuitry.

Interrupt on Non-Correctable Error Mode — In this mode the RAM board will interrupt the iSBC processor board only when a non-correctable (multiple bit) error is detected by the ECC circuitry. A multiple bit error is not correctable by the ECC circuitry.

Correcting Mode — In this mode the RAM board corrects any correctable error (single bit error). Errors which are not correctable are not modified. Interrupts are generated depending on the interrupt mode selected.

Non-Correcting Mode — In this mode the RAM board does not correct any error. The ECC circuitry continues to check for errors, but no corrective action is taken. Interrupts continue as described previously.

Diagnostic Mode — This mode is used for testing the on-board ECC circuitry. In this mode the write enable strobe to the ECC RAM array is continuously disabled. The diagnostic mode can be used to simulate errors and in conjunction with the "Examine Syndrome Word Mode" examine the check bits generated by the ECC circuitry.

Examine Syndrome Word Mode — This mode, in conjunction with the diagnostic mode, is used for testing the ECC memory. In this mode, the syndrome bits/check bits are clocked into the ESR on every memory read/write cycle, respectively. The ESR translation PROM switches to a transparent mode in the examine syndrome word mode. This allows the actual syndrome word generated by the 8206 ECC device to be examined.

ERROR STATUS REGISTER

This 8-bit register contains information about memory errors. The ESR reflects the latest error occurence. Table 1 shows the status register format. Bits 5 and 6 show the failing row while bits 0 through 4 indicate which bit (of the 16-bit data word or the 6-bit ECC syndrome) is in error. Bit 7 is always high.

Table 1. Error Status Register Format

Bit 6	Bit 5	Meaning	
0	0	Error in row	0
0	1		1
1	0		2
1	1		3

Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Meaning	
0	0	0	0	0	Error in data bit	0
0	0	0	0	1		1
0	0	0	1	0		2
0	0	0	1	1		3
0	0	1	0	0		4
0	0	1	0	1		5
0	0	1	1	0		6
0	0	1	1	1		7
0	1	0	0	0		8
0	1	0	0	1		9
0	1	0	1	0	Error in data bit	10
0	1	0	1	1		11
0	1	1	0	0		12
0	1	1	0	1		13
0	1	1	1	0		14
0	1	1	1	1		15
1	0	0	0	0	Error in check bit	0
1	0	0	0	1		1
1	0	0	1	0		2
1	0	0	1	1		3
1	0	1	0	0		4
1	0	1	0	1		5
1	1	1	1	0	No Error	
1	1	1	1	1	Non-correctable (multiple-bit error)	

Battery Back-up/Memory Protect

An auxillary power bus is provided to allow separate power to the RAM array for systems requiring backup of read/write memory. An active low TTL compatible memory protect signal is brought out on the auxillary bus connector which, when asserted, disables read/write access to the RAM board. This input is provided for the protection of RAM contents during system power-down sequences.

SPECIFICATIONS

Word Size Supported

8- or 16-bits

Memory Size

131,072 bytes (iSBC 028CX board)
262,144 bytes (iSBC 056CX board)
524,288 bytes (iSBC 012CX board)
1,048,576 bytes (iSBC 010CX board)
2,097,152 bytes (iSBC 020CX board)

Access Times (All densities)

MULTIBUS® System Bus

Read/Full Write — 380 ns (max)

Write Byte — 530 ns (max)

iLBX™ Local Bus

Read/Full Write — 340 ns (max)

Write Byte — 440 ns (max)

Cycle Times (All densities)

MULTIBUS® System Bus

Read/Full Write — 490 ns (max)

Write Byte — 885 ns (max)

iLBX™ Local Bus

Read/Full Write — 375 ns

Write Byte — 740 ns

NOTE: If an error is detected, read access time and cycle times are extended to 255 ns (max)

Memory Partitioning

Maximum System memory size is 16M Bytes for both MULTIBUS and iLBX BUS. MULTIBUS partitioning is by Page, Block and Base, while the iLBX BUS is by Block and Base only.

Page Address

MULTIBUS® — 0-4 megabytes; 4-8 megabytes, 8-12 megabytes; 12-16 megabytes

iLBX™ BUS — N/A

Base Address

MULTIBUS® System Bus — Any 16K byte boundary within the 4M-byte page.

iLBX™ Local Bus — Any 64K byte boundary selectable on board boundaries to 8M-bytes and some 64K-byte boundaries in the first megabyte. Others available if PAL programming is changed.

Power Requirements

Voltage — 5 VDC ± 5%

Product	Current	Standby (Battery Back-up)
iSBC® 028CX Board	3.8A (typ.) 6.5A (max.)	2.0A (typ.) 2.1A (max.)
iSBC® 056CX Board	4.0A (typ.) 6.6A (max.)	2.1A (typ.) 2.2A (max.)
iSBC® 012CX Board	4.4A (typ.) 6.8A (max.)	2.2A (typ.) 2.4A (max.)
iSBC® 010CX Board	4.8A (typ.) 7.0A (max.)	2.1A (typ.) 2.3A (max.)
iSBC® 020CX Board	5.3A (typ.) 7.5A (max.)	2.2A (typ.) 2.4A (max.)

Environmental Requirements

Operating Temperature — 0°C to 55°C airflow of 200 linear feet per minute

Operating Humidity — To 90% without condensation

Physical Dimensions

Width — 30.48 cm (12 inches)

Height — 17.15 cm (6.75 inches)

Thickness — 1.27 cm (0.50 inches)

Weight — iSBC 028CX board: 4699 gm (16.7 ounces); iSBC 056CX board: 5329 gm (19.0 ounces); iSBC 012CX board: 6589 gm (23.5 ounces); iSBC 010CX board: 5329 gm (19.0 ounces); iSBC 020CX board: 6589 gm (23.5 ounces)

Reference Manuals

145158-003 — iSBC® 028CX/iSBC® 056CX/iSBC® 012CX Hardware Reference Manual

144456-001 — Intel iLBX™ 010CX, 020CX Specification

9800683-03 — Intel MULTIBUS® Specification

Manuals may be ordered from any Intel Sales Representative, Distributor Office or from the Intel Literature Department, 3065 Bowers Avenue, Santa Clara, CA. 95051

ORDERING INFORMATION

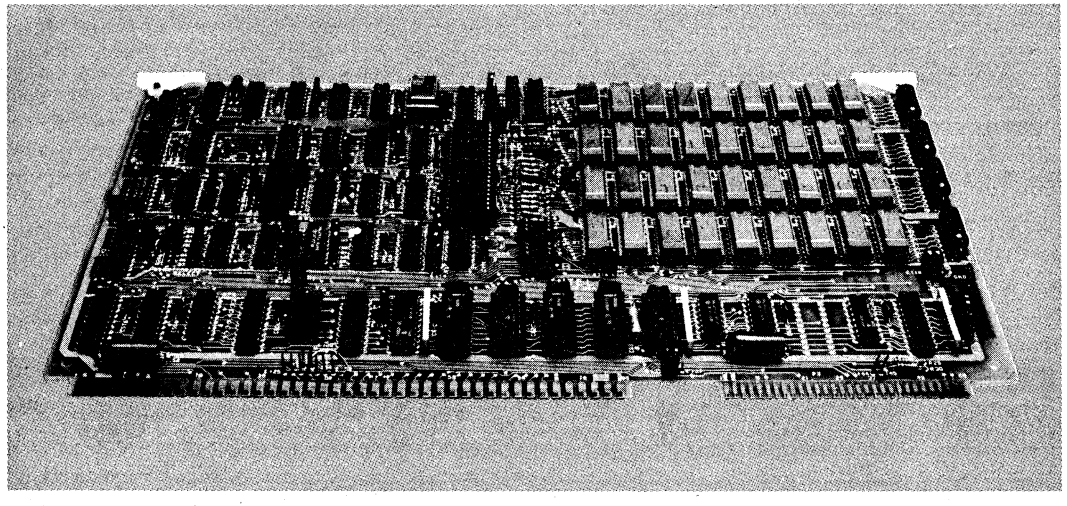
Part Number	Description
iSBC® 012CX	512K byte RAM board with ECC and iLBX™ Connectors
iSBC® 056CX	256K byte RAM board with ECC and iLBX™ Connectors
iSBC® 028CX	128K byte RAM board with ECC and iLBX™ Connectors
iSBC® 010CX	1M byte RAM board with ECC and iLBX™ Connectors
iSBC® 020CX	2M byte RAM board with ECC and iLBX™ Connectors

iSBC® 028A/056A
RAM MEMORY BOARDS

- **iSBC®86, iSBC®88 and iSBC®80 board RAM expansion through direct MULTIBUS® interface**
- **128K or 256K bytes of read/write memory**
- **On-board parity generator/checker and error status register**
- **Requires a single +5 volt power supply**
- **Assignable anywhere within a 16 megabyte address space**
- **Jumper selectable base address on any 4K byte boundary**
- **Auxiliary power bus and memory protect control logic for battery backup RAM requirements**

The iSBC® 028A and iSBC 056A RAM memory boards are members of Intel's complete line of iSBC memory and I/O expansion boards. Each board interfaces directly to any iSBC 80, iSBC 88 or iSBC 86 Single Board Computer via the MULTIBUS® interface to expand system RAM capacity. The iSBC 028A and iSBC 056A boards contain 128K, or 256K bytes of read/write memory implemented using dynamic RAM components. An on-board LSI dynamic RAM controller refreshes a portion of these components every 14 microseconds. Each refresh cycle utilizes memory for 480 nanoseconds (maximum).

The iSBC 028A and iSBC 056A boards generate byte oriented parity during all write operations and perform parity checking during all read operations. When a parity error is detected, these boards can generate an interrupt on the MULTIBUS interface. In addition, the row and bank of the RAM array containing the error are stored in a Parity Flag Register (see Figure 1). This register is accessible as a MULTIBUS I/O port. An on-board LED also provides a visual indication that a parity error has occurred. To facilitate testing of these boards, parity generation and checking can be changed from even to odd under software control.



OCTOBER 1984
ORDER NUMBER: 143877-002

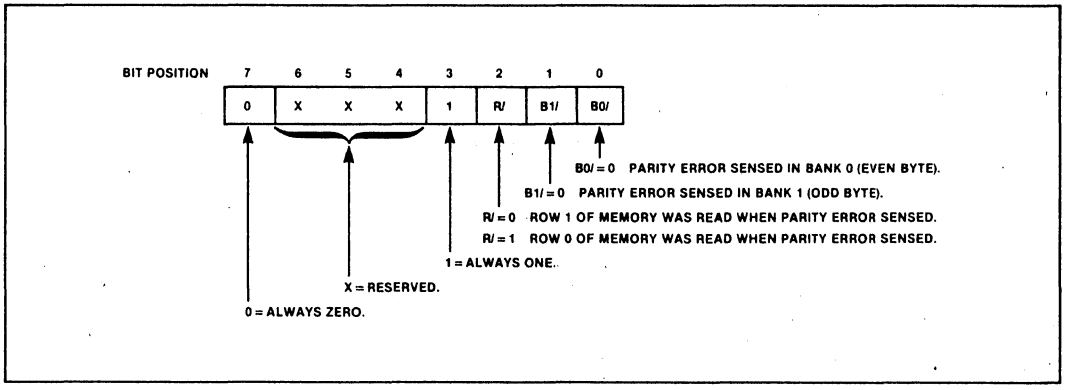


Figure 1. Parity Flag Register Format

SPECIFICATIONS

Word Size

8 bits and 16 bits

Memory Size

131,072 bytes (iSBC 028A); or 262,144 bytes (iSBC 056A)

Access Time

iSBC 028A

500 ns max. (worst case)
460 ns max. (typical)

iSBC 056A

570 ns max. (worst case)
530 ns max. (typical)

Cycle Times (Worst Case)

Read

iSBC 028A — 600 ns max.
iSBC 056A — 650 ns max.

Write

iSBC 028A — 600 ns max.
iSBC 056A — 650 ns max.

Refresh

iSBC 028A — 480 ns max.
iSBC 056A — 600 ns max.

Interface

All address, data and command signals are TTL compatible.

Address Selection

Memory — Base address is jumper selectable on any 4K byte boundary in a 16 megabyte address space. On-board RAM cannot cross a megabyte address boundary.

Parity Flag Register — The I/O address of the Parity Flag Register is jumper selectable to be between 00H to 0FH or 40H to 4FH.

Connector

Edge connector — 86 pin double-sided PC edge connector with 0.156 in. contact centers.

Mating connector — Viking 3KH43/9AMK12 or equivalent.

Auxiliary Power

An auxiliary power bus is provided to allow separate power to RAM array for systems requiring battery backup of read/write memory. Selection of this auxiliary RAM power bus is made via jumpers on the board.

Memory Protect

An active-low TTL compatible memory protect signal is brought out on the auxiliary connector which, when asserted, disables read/write access to RAM on the board. This input is provided for the protection of RAM contents during system power-down sequences.

Physical Characteristics

Width — 12.00 in. (30.48 cm)
Height — 6.75 in. (17.15 cm)
Depth — 0.50 in. (1.27 cm)
Weight — 14 oz. (397 gm)

Electrical Characteristics

D.C. POWER REQUIREMENTS

All configurations require only +5 volts ±5%.

Normal System Operation (max.)
iSBC 028A/056A — 4.57A (worst case)
3.66A (typical)

Auxiliary Power No RAM Access (max.)
iSBC 028A/056A — 0.55A (worst case)
0.45A (typical)

Environmental Characteristics

Operating Temperature — 0°C to +55°C
Relative Humidity — to 90% (without condensation)

Reference Manual

143572-001 — iSBC 032A/064A/028A/056A Hardware Reference Manual (not supplied)

Manuals may be ordered from any Intel sales representative, distributor office, or from Intel Literature Dept., 3065 Bowers Avenue, Santa Clara, California 95051.

ORDERING INFORMATION

Part Number	Description
SBC 028A	128K-Byte RAM Board with Parity
SBC 056A	256K-Byte RAM Board with Parity

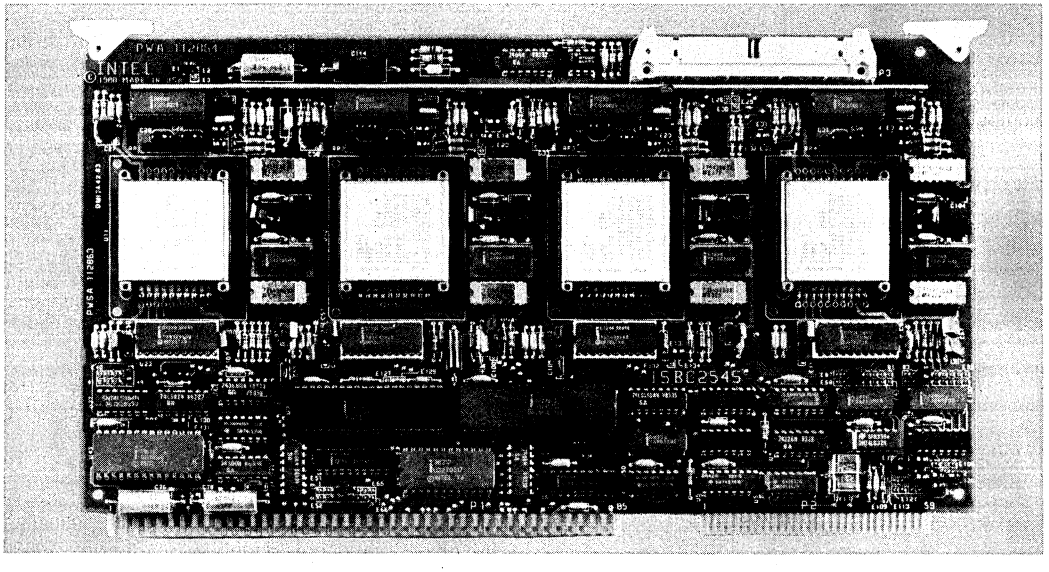
iSBC® 254S
BUBBLE MEMORY BOARD

- **Capacity up to 512K Bytes of Bubble Memory Storage**
- **Automatic Error Correction Capability**
- **Operates from Standard +5V and +12V Power Supplies**
- **High-Density Storage**
- **DMA Capability**
- **Non-Volatile Storage**
- **High Reliability Even Under Harsh and Rugged Environments**
- **Average Access Time of 48ms**
- **Burst Data Rate up to 200K Bytes per Second**
- **Software Compatible with the iRMX™ Operating System**
- **Powerfail Data Protection**

The iSBC 254S board is a completely assembled and tested non-volatile read/write memory utilizing the Intel 7110 one-megabit bubble memory. This board is offered with one, two, or four 7110 bubble memories, thus yielding capacities of 128K, 256K, or 512K bytes.

Software support is provided under both iRMX/80 and iRMX/86 operating systems, and DMA capability provides the user with considerable flexibility and control. Because of the solid-state nature of this technology, the iSBC 254S board is ideally suited for applications in harsh or rugged environments.

The iSBC 254S board is compatible with 16-bit addressing for 8-bit processors and with 20-bit addressing for 16-bit processors.



MARCH 1983
ORDER NUMBER: 210802-002

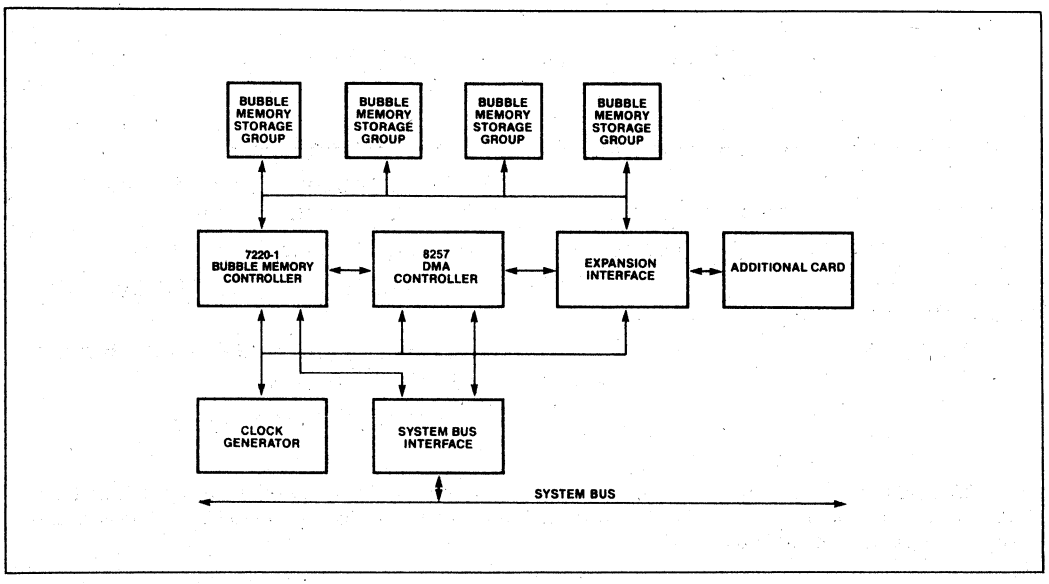


Figure 1. iSBC® 254S Board, Block Diagram

SPECIFICATIONS

Memory Size

128K, 256K, or 512K bytes

Interface

All address, data, and control signals are TTL-compatible and Intel MULTIBUS® system compatible.

Performance

Maximum Data Rate: 200K bytes/second
Average Access Time: 48ms
Power Supply Requirements

Electrical Characteristics

D.C. Power, supplied through MULTIBUS® connector

Voltage	Tolerance	Power Off/Power Fail Decay Rate	Max. Current
+12 Volts	±5%	less than 1.10 volts/msec	1.4A
+5 Volts	±5%	less than 0.45 volts/msec	3.0A

- Voltage sequencing—no restrictions
- Power on voltage rate of rise—no restrictions
- The power supply requirements shown are based on the recommended power fail circuitry.

Connector

86-pin double-sided PC edge connector with 0.40 cm (0.156 in.) contact centers.
Mating Connector: Control Data VFB01E43D0A1 or Viking 2VH43/1ANE5.

Physical Characteristics

Length: 30.48 cm (12 in.)
Height: 17.15 cm (6.75 in.)
Depth: 1.57 cm (0.62 in.)

Note: Because of its depth, the iSBC 254S board requires two card slots in standard MULTIBUS card frame.

Environment

Board Ambient Operating Temperature: 0° to 55°C
Non-Volatile Storage Temperature: – 40° to 90°C

Additional Documentation

iSBC 254S Reference Manual (Order No. 113844)
INTEL MULTIBUS Specification (Order No. 9800683)
Memory Components Handbook (Order No. 210830)

OPERATIONAL DESCRIPTION

Like many high density peripheral storage devices, bubble memory data is serially organized into pages rather than bytes. A page, varying in length from 64 bytes (one bubble on board with error correction) to 512 bytes (four bubbles on board with error correction) is the smallest increment of data that can be transferred. A file consisting of a few large pages can be transferred faster than an equivalent file formatted with smaller size pages. The iSBC 254S is offered with one, two, or four 7110 bubble memories which provide the user with the flexibility to match performance, density, and cost to the application.

Data transfers are accomplished under software control by the 7220-1 bubble memory controller. Three distinct I/O modes of operation relate to the transfer of data. Choice of mode can be tailored to each user's application. Each I/O mode is briefly described below.

I/O Modes for Data Transfer

DMA (Direct Memory Access) I/O Mode is the highest performance mode of data transfer. An on-board INTEL 7220-1 Bubble Memory Controller (BMC) and INTEL 8257 DMA controller work in conjunction to generate the MULTIBUS handshake protocol signals. Once a data block transfer begins in this mode, CPU involvement is not required until the entire transfer is completed. This frees the CPU to perform other tasks during DMA transfers. A single data block transfer can be up to 16K bytes in length.

Interrupt I/O Mode requires moderate CPU involvement for data transfers. The BMC generates an on-board interrupt signal when the BMC FIFO (First In-First Out) buffer is either half full (bubble read operation) or half empty (bubble write operation). The interrupt signal is translated to MULTIBUS interrupt line via a jumper. Using this interrupt-driven scheme, software is responsible for performing the appropriate transfer of data (typically 22 bytes) to or from the FIFO buffer when the interrupt occurs.

Polled I/O Mode is the most simple. However, it is also the most demanding of CPU time. System software must determine when to transfer data to or from the FIFO by continually polling a status bit in the BMC status register. The status bit indicates presence or absence of data in the FIFO on a byte-by-byte basis.

COMMAND EXECUTION

A set of 16 separate commands can be issued to the BMC on the iSBC 254S for complete system control of the bubble memory. Command execution can be viewed as divided into two phases: command execution and result. The software responsibilities for the command execution phase consist of loading the BMC parametric registers (command dependent), issuing the desired command, and then verifying that the command was accepted. The appropriate data transfer technique then would transfer data as required. The result phase consists of the system software determining the completion status (successful or unsuccessful) of the command by either polling the BMC status register (polled) or through an interrupt service routine (interrupt).

SOFTWARE DESCRIPTION

The iSBC 254S board can run under either the iRMX/80 or the iRMX/86 operating system.

Under the iRMX/80 operating system a set of two iRMX/80 software tasks perform data transfers. Bubble I/O (BUBIO) provides the interface routines for data storage and retrieval. A second task, the Bubble Manager (BMGR) keeps track of free or available space on the bubble memory device. BMGR operates very similar to the iRMX/80 free space manager by allocating portions of bubble memory space at the request of a user task. BUBIO can be configured to run independent of BMGR.

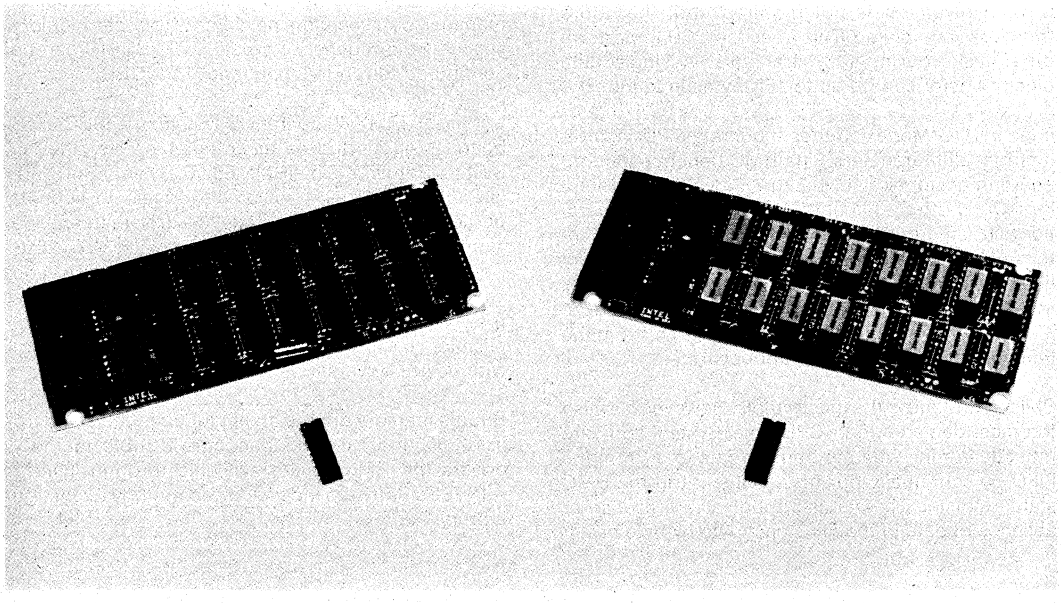
Under the iRMX/80 operating system, the iSBC 254S board is supported as an integral part of the I/O System Software. The iRMX 86 I/O System Software is implemented as a set of file drivers to support particular types of files and device drivers to provide support to particular devices (i.e., the iSBC 254S Board). Each type of file has its own file driver and each device has its own device driver. This provides great flexibility and device independence since application tasks communicate with file drivers, not with device drivers.

Bubble memory drivers are included in the standard iRMX/86 and iRMX/88 package. Bubble memory drivers for iRMX 80 are available through Insite™. —INTEL's Software Index and Technology Exchange Library.

iSBC® 304 128K BYTE RAM MULTIMODULE™ BOARD
iSBC® 300A 32K BYTE RAM MULTIMODULE™ BOARD

- **iSBC® 304 module provides 128K bytes of dual port RAM expansion for the iSBC® 86/30 or iSBC® 86/35 board**
- **iSBC® 300A module provides 32K bytes of dual port RAM expansion for the iSBC® 86/14 board**
- **Simple, reliable, mechanical and electrical interconnection**
- **On-board memory expansion for the iSBC® 86/30, iSBC® 86/14 and iSBC® 86/35 Single Board Computers**
- **On-board memory expansion eliminates MULTIBUS® system bus latency and increases system throughput**
- **Low power requirements**

The iSBC® 304 and iSBC 300A RAM modules provide simple, low cost expansion of the memory compliment available on the iSBC 86/30 and iSBC 86/14 Single Board Computers, respectively. Each module doubles the on-board RAM memory capacity of the host board. Additionally, the iSBC 304 provides 128K bytes ram expansion to the iSBC 86/35 giving a total capacity of 640K bytes ram memory. The RAM MULTIMODULE options for the host boards offer system designers a new level of flexibility in defining and implementing Intel single board computer systems. Because they expand the memory configuration on-board, they can be accessed as quickly as the existing host board memory by eliminating the need for accessing the additional memory via the MULTIBUS system bus.



January, 1982
Order Number: 210329-001

FUNCTIONAL DESCRIPTION

Each MULTIMODULE contains dynamic RAM devices and sockets for the Intel 8203 dynamic RAM controller and memory interface latching. To install the module, the latches and controller from the host CPU board are removed and inserted into sockets on the RAM MULTIMODULE. The module is then mounted onto the host board. Pins extending from the controller and latch sockets mate with device sockets underneath (see Figure 1). Additional pins mate to supply other signals to complete the electrical interface.

The module is then secured at three additional points with nylon hardware to ensure the mechanical security of the assembly.

To complete the installation, one socketed PROM is replaced on the host CPU board with the one supplied with the MULTIMODULE kit. This is the MULTIBUS address decode PROM which allows the host board logic to recognize its expanded onboard memory compliment.

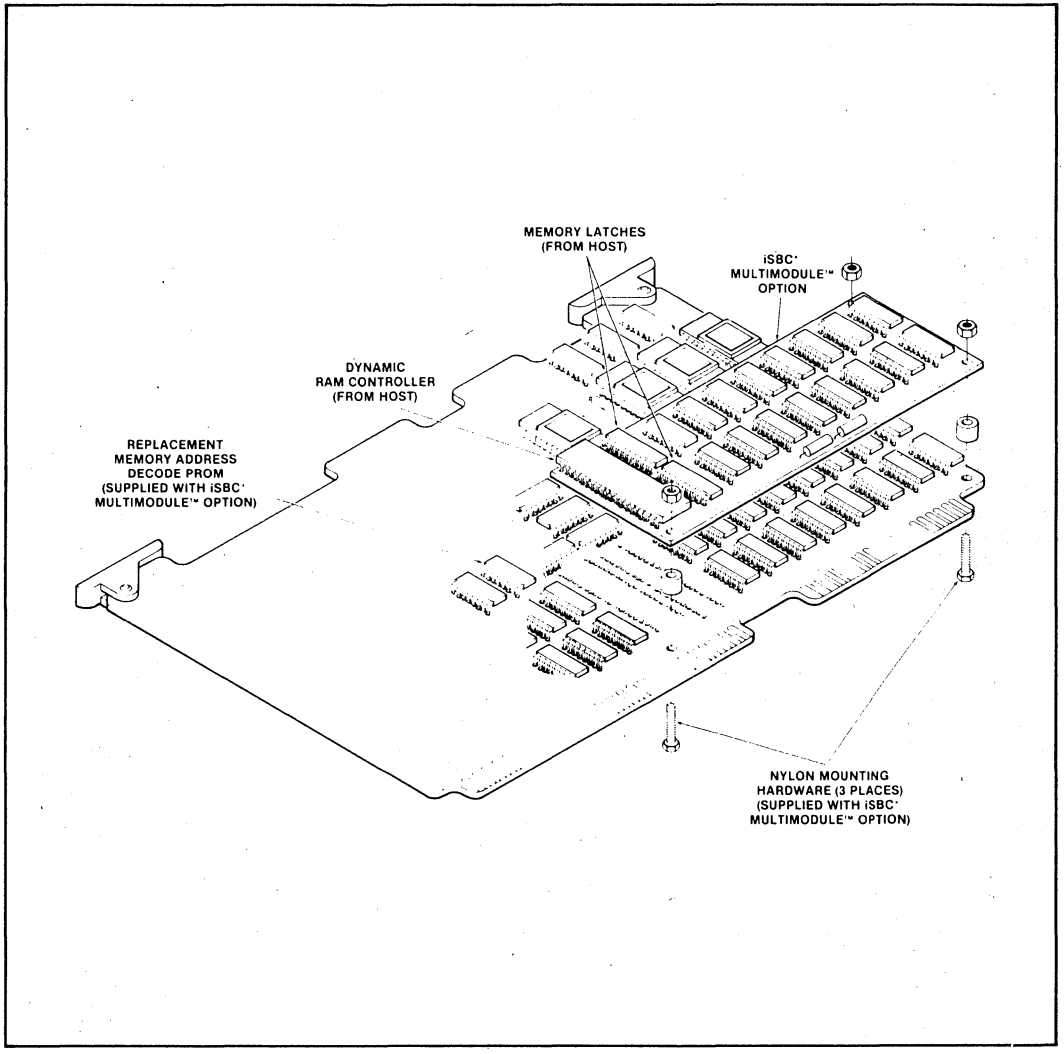


Figure 1. Installation of the MULTIMODULE™ RAM on the Host Single Board Computer

SPECIFICATIONS

Word Size

8 or 16 bits (16-bit data paths)

Memory Size

iSBC® 304 Module — 128K bytes RAM

iSBC® 300A Module — 32K bytes RAM

Cycle Time

iSBC® 304 — 700 nsec (read); 700 nsec (write)

iSBC® 300A — 700 nsec (read); 700 nsec (write)

Memory Addressing

CPU ACCESS

iSBC® 304 (with iSBC® 86/35) — 640K bytes (total capacity); 0-9FFFF$_H$ (address range)

iSBC® 304 (with **iSBC® 86/30**) — 256K bytes (total capacity); 0-3FFFFH (address range)

iSBC® 300A (with **iSBC® 86/14**) — 64K bytes (total capacity); 0-0FFFFH (address range)

MULTIBUS® Access

Jumper selectable for any 32K (8K) byte boundary, but not crossing a 256K (128K) byte boundary on the iSBC 86/30 (iSBC 86/14) host board.

Interface

The interfaces for the iSBC 304 and iSBC 300A module options are designed only for the iSBC 86/30 and iSBC 86/14 host boards, respectively.

Private Memory Allocation

Segments of the combined host/MULTIMODULE RAM memory may be configured as a private resource, protected from MULTIBUS system access. The amount of memory allocated as a private resource may be configured in increments of 25% of the total on-board memory ranging from 0% to 100%. The iSBC 304 module mounted on the iSBC 86/30 board, therefore, supports private allocation of 64K, 128K, 192K, or 256K bytes of RAM memory. The iSBC 300A module mounted on the iSBC 86/14 board supports private allocation of 16K, 32K, 48K, or 64K bytes of RAM memory.

Auxiliary Power

The low power memory protection option included on the CPU host boards supports the RAM modules.

Physical Characteristics

Width — 2.4 in. (6.10 cm)

Height — 5.75 in. (14.61 cm)

Depth* — .72 in. (1.83 cm)

Weight — .13 oz. (59 g)

* Note: Combined depth including host board.

Electrical Characteristics

DC POWER REQUIREMENTS

iSBC® 304 — 640 ma at +5 volts incremental power

iSBC® 300A — 256 ma at +5 volts incremental power

Environmental Characteristics

Operating Temperature — 0°C to 55°C

Relative Humidity — to 90% (without condensation)

Reference Manual

All necessary documentation for the iSBC 304 and iSBC 300A MULTIMODULE boards is included in the iSBC 86/14 and iSBC 86/30 Hardware Reference Manual (NOT SUPPLIED).

Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, CA 95051.

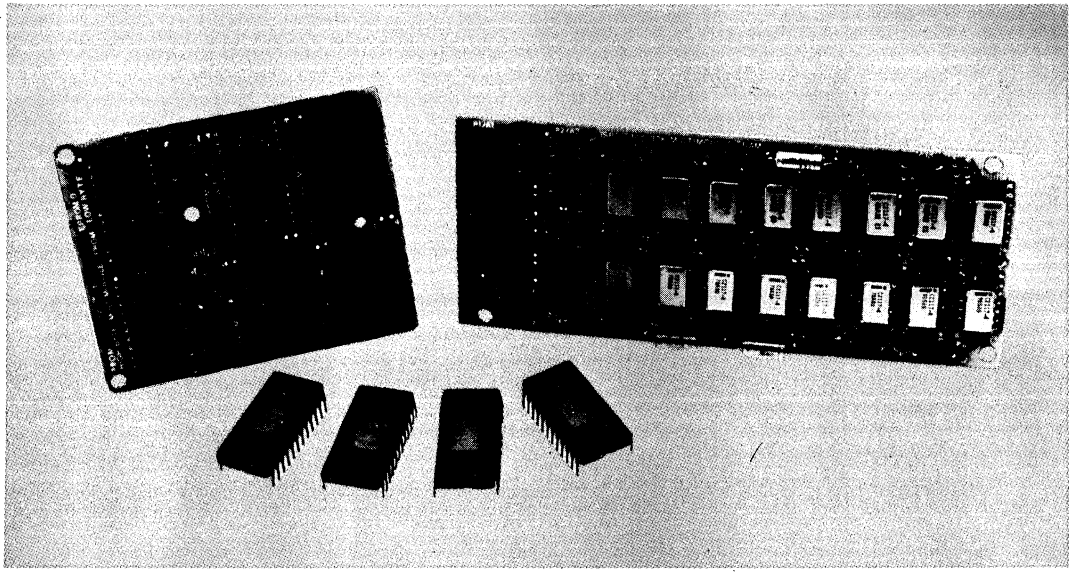
ORDERING INFORMATION

Part Number	Description
SBC 304	128K MULTIMODULE option for iSBC 86/30 or iSBC 86/35 cpu boards
SBC 300A	32K MULTIMODULE option for iSBC 86/14 board

iSBC® 300 or (pSBC 300*)
32K-BYTE RAM EXPANSION MODULE
iSBC® 340 or (pSBC 340*)
16K-BYTE EPROM EXPANSION MODULE

- **On-board memory expansion for iSBC® 86/12A Single Board Computer**
- **iSBC®300 module provides 32K bytes of dual port dynamic RAM and plugs directly into the iSBC®86/12A board**
- **iSBC®340 module provides sockets for up to 16K bytes of additional EPROM and plugs directly into the iSBC 86/12A board**
- **On-board memory expansion eliminates MULTIBUS®system bus latency and increases system throughput**
- **Low power requirements**
- **Simple, reliable mechanical and electrical interconnection**

The iSBC 300 32K-byte RAM expansion module and the iSBC 340 16K-byte EPROM expansion module provide simple, low cost expansion of the memory complement available on the iSBC 86/12A single board computer. Each module utilized individually or together can double the iSBC 86/12A board's on-board RAM and EPROM memory capacity. The iSBC 300 32K-byte RAM expansion module and the iSBC 340 16K-byte EPROM expansion module options for the iSBC 86/12A board offer system designers a new level of flexibility in defining and implementing Intel® single board computer systems. These options allow the systems designer to double the memory complement of an iSBC 86/12A board with a minimum of system implications. Because they expand the memory configuration on-board, they can be accessed as quickly as the existing iSBC 86/12A memory by eliminating the need for accessing the additional memory via the MULTIBUS system bus. With the iSBC 86/12A board mounted in the top slot of an iSBC 604 or iSBC 614 cardcage, sufficient clearance exists for mounting both the iSBC 300 and/or the iSBC 340 expansion module option(s). If the iSBC 86/12A board is inserted into some other slot, the combination of boards will physically (but not electrically) occupy two cardcage slots. Incremental power required by the options is minimal; for instance, only 305 mW is needed for the iSBC 300 RAM expansion module.



*Same product, manufactured by Intel Puerto Rico, Inc.

FUNCTIONAL DESCRIPTION

iSBC® 300 32K-Byte MULTIMODULE ™ RAM

The iSBC 300 module contains sixteen 16K-byte dynamic RAM devices, sockets for the Intel® 8202A Dynamic computer. It expands the iSBC 86/12A board's on-board dual port RAM capacity from 32K bytes to 64K bytes. The iSBC 300 module contains sixteen 16K-byte dynamic RAM devices, sockets for the Intel® 8202 Dynamic RAM Controller and memory interface latching. To install the iSBC 300 module, the latches and controller from the iSBC 86/12A board are removed and inserted into the sockets on the iSBC 300 module. The add-on board is then mounted onto the iSBC 86/12A board. Pins extending from the controller's and latches' sockets mate with the devices' sockets underneath (see Figure 1). Additional pins mate to supply power and other signals to complete the electrical interface. The module is then secured at three additional points with nylon hardware to insure the mechanical security of the assembly.

To complete the installation, two socketed PROMs are replaced on the iSBC 86/12A board with those supplied with the iSBC 300 kit. These are the on-board memory and MULTIBUS address decode PROMs which allow the iSBC 86/12A board logic to recognize its expanded on-board memory complement.

iSBC® 340 16K-byte MULTIMODULE ™ EPROM

The iSBC 340 module expands the iSBC 86/12A Single Board Computer's on-board EPROM capacity from 16K bytes to 32K bytes. It measures 3.3" by 2.8" and consists of a PC board with six 24-pin special sockets. Two of the sockets have extended pins which mate with two of the EPROM sockets on the iSBC 86/12A board. Two of the EPROMs which would have been inserted on the iSBC 86/12A board are then reinserted in the iSBC 340 module. Additional pins also mate for bringing chip selects for the remaining EPROM devices (see Figure 2). The mechanical interface is similar to that used on the iSBC 300 RAM module and consists of two additional mounting holes and the necessary mounting hardware.

The iSBC 340 module supports Intel® 2732A EPROM. One section of the iSBC 86/12A on-board memory and MULTIBUS address decode PROMs (the same decode PROMs mentioned for the iSBC 300 module) is already preprogrammed to support the iSBC 340 module with Intel® 2732A EPROMs. This section is selected through the EPROM configuration switches on the iSBC 86/12A board. The iSBC 340 board can optionally be configured by the user to support Intel® 2758 or 2761 EPROMs by programming new iSBC 86/12A decode PROMs to support these devices. Necessary documentation and PROM map listings are in the iSBC 86/12A Hardware Reference Manual (order number 9803074-01).

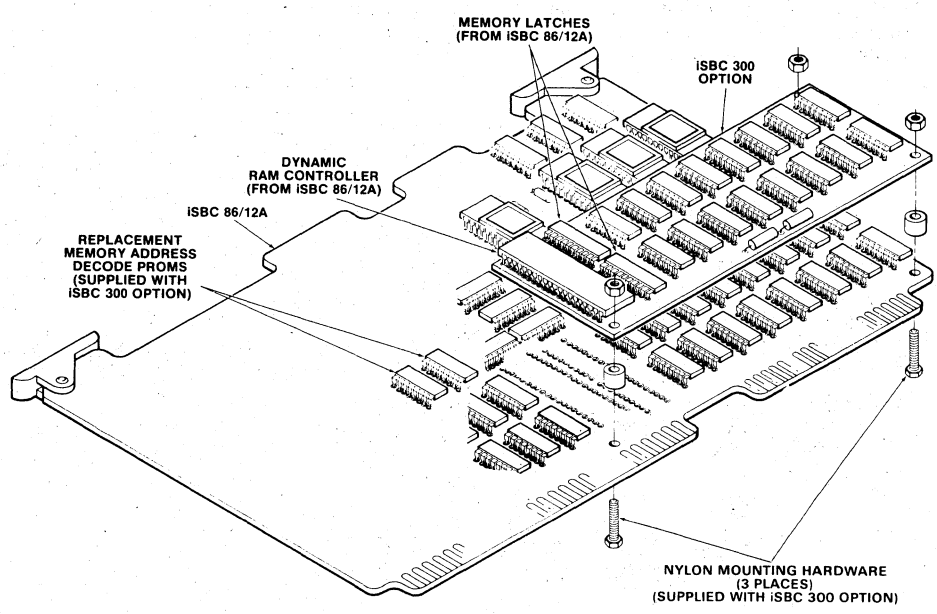


Figure 1. Installation of iSBC® 300 MULTIMODULE ™ RAM on iSBC® 86/12A Single Board Computer

AFN-01282A

SPECIFICATIONS

Word Size
8 or 16 bits (16-bit data paths)

Memory Size
iSBC 300 Module — 32,768 bytes of RAM
iSBC 340 Module — 16,384 bytes (max) of EPROM

Access Time
iSBC 300 Module — Read: 1 μsec, write: 1.2 μsec
iSBC 340 Module — Standard EPROMs (450 nsec): 1 μsec, fast EPROMs (350 or 390 nsec): 800 nsec

Interface
The interface for the iSBC 300 and iSBC 340 module options is designed only for Intel's iSBC 86/12A Single Board Computer.

Memory Addressing
On-board RAM

CPU Access

iSBC 86/12A board only (32K bytes) — 00000-07FFFH.
iSBC 86/12A board + iSBC 300 module (64K bytes) — 00000-0FFFFH.
MULTIBUS Access — Jumper selectable for any 8K-byte boundary, but not crossing a 128K-byte boundary.

On-board EPROM

iSBC 86/12A board only (16K-bytes max.) — FF000-FFFFFH (using 2758 EPROMs); FE000-FFFFFH (using 2316E ROMs or 2716 EPROMs); and FC000-FFFFFH (using 2332A ROMs or 2732A EPROMs).

iSBC 86/12A board + iSBC 340 module (32K bytes max) — FE000-FFFFFH (using 2758 EPROMs); FC000-FFFFFH (using 2716 EPROMs); F8000-FFFFFH (using 2732A EPROMs).

On-board EPROM/ROM is not accessible via the MULTIBUS interface.

Auxiliary Power/Memory Protection
The low power memory protection option included on the iSBC 86/12A boards supports the iSBC 300 RAM module.

"Local Only" Memory Protection
The iSBC 86/12A Single Board Computer supports dedication of on-board RAM for on-board CPU access only in 8K, 16K, 24K, or 32K-byte segements. Installation of the iSBC 300 option allows protection of 16K, 32K, 48K, or 64K-byte segments.

Physical Characteristics

	iSBC 300	iSBC 340
Width	5.75"	3.3"
Length	2.35"	2.8"
Height of iSBC 86/12A plus mounted option	.718	.718*
Weight	13 oz.	5 oz.

*Includes EPROMs

All necessary mounting hardware (nylon, screws, spacers, nuts) are supplied with each kit.

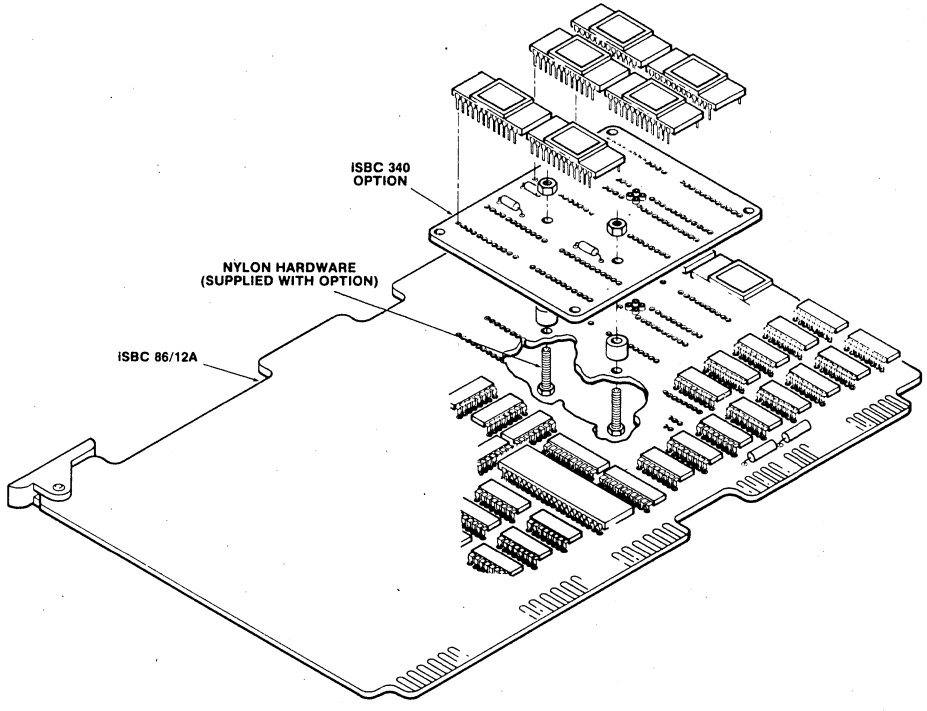


Figure 2. Installation of iSBC® 340 MULTIMODULE ™EPROM Option on iSBC® 86/12A Single Board Computer

AFN-01282A

Electrical Characteristics

DC power requirements:

Voltage	iSBC 300	iSBC 340
+5 ±5%	1 mA	120 mA[1]
+12 ±5%	24 mA	—
-12 ±5%	1 mA	—

Note:
1. Loaded with Intel 2732A EPROMs.

Environmental Characteristics

Operating Temperature — 0° to +55°C
Relative Humidity — to 90% (without condensation)

Reference Manuals

All necessary documentation for the iSBC 300 MULTIMODULE RAM and iSBC 340 MULTIMODULE EPROM/ROM is included in the iSBC 86/12A Hardware Reference Manual; order #9803074-01. (NOT SUPPLIED)

Manuals may be ordered from any Intel sales representative distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, CA 95051.

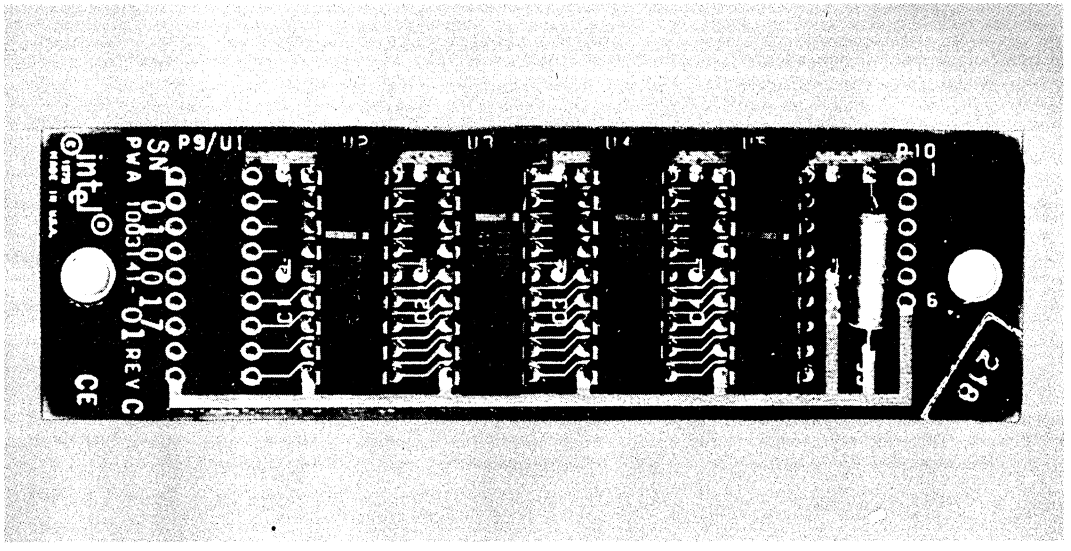
ORDERING INFORMATION

Part Number	Description
SBC 300	32K byte MULTIMODULE RAM
SBC 340	16K byte MULTIMODULE EPROM

iSBC® 301
4K-BYTE RAM
MULTIMODULE™ BOARD

- On-board memory expansion to 8K bytes for iSBC® 80/24 and iSBC® 88/40 Single Board Computers
- Provides 4K bytes of static RAM directly on-board
- Uses 5 MHz (8185-2) RAMs
- Single +5V supply
- 0.5 watts incremental power dissipation
- On-board memory expansion eliminates MULTIBUS® system bus latency and increases system throughput
- Reliable mechanical and electrical interconnection

The Intel iSBC 301 4K-Byte RAM MULTIMODULE Board provides simple, low cost expansion to double the RAM capacity on the iSBC 80/24 or iSBC 88/40 Single Board Computer to 8K bytes. This offers system designers a new level of flexibility in defining and implementing system memory requirements. Because memory is configured on-board, it can be accessed as quickly as the existing iSBC 80/24 or iSBC 88/40 memory, eliminating the need for accessing the additional memory via the MULTIBUS system bus. As a result, the iSBC 301 board provides a high speed, cost effective solution for systems requiring incremental RAM expansion. Incremental power required by the iSBC 301 module is minimal, dissipating only 0.5 watts.



FUNCTIONAL DESCRIPTION

The iSBC 301 board measures 3.95″ by 1.20″ and mounts above the RAM area on the iSBC 80/24 or iSBC 88/40 single board computer. It expands the on-board RAM capacity from 4K bytes to 8K bytes. The iSBC 301 MULTIMODULE board contains four 1K byte static RAM devices and a socket for one of the RAM devices on the iSBC 80/24 or iSBC 88/40 board. To install the iSBC 301 MULTIMODULE board, one of the RAMs is removed from the host board and inserted into the socket on the iSBC 301 board. The add-on board is then mounted into the vacated RAM socket on the host board. Pins extending from the RAM socket mate with the device's socket underneath (see Figure 1). Additional pins mate to the power supply and chip select lines to complete the electrical interface. The MULTIMODULE board is then secured at two additional points with nylon hardware to insure mechanical security of the assembly. With the iSBC 80/24 or iSBC 88/40 board mounted in the top slot of an iSBC 604 or iSBC 614 cardcage, sufficient clearance exists for mounting the iSBC 301 option. If the iSBC 80/24 or iSBC 88/40 board is inserted into some other slot, the combination of boards will physically (but not electrically) occupy two cardcage slots.

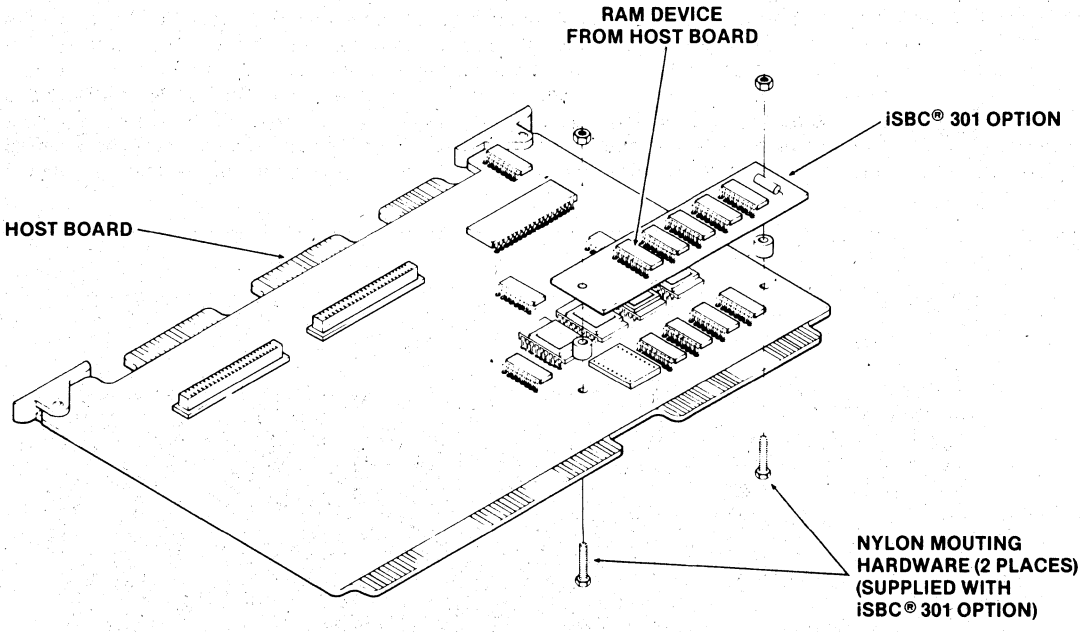


Figure 1. Installation of iSBC® 301 4K-Byte RAM MULTIMODULE™ Board

AFN-01487A

SPECIFICATIONS

Word Size
8 bits

Memory Size
4096 bytes of RAM

Access Time
Read: 140 ns (from READ command)
200 ns (from ALE)
Write: 150 ns (from READ command)
190 ns (from ALE)

Memory Addressing
Memory addressing for the iSBC 301 4K-Byte RAM MULTIMODULE Board is controlled by the host board via the address and chip select signal lines and is contiguous with the host board RAM.

iSBC 80/24 and iSBC 301 board: 02000-02FFF
iSBC 88/40 and iSBC 301 board: 00000-01FFF

Physical Characteristics
Width — 1.20 in. (3.05 cm)
Length — 3.95 in. (10.03 cm)
Height — .44 in. (1.12 cm) iSBC 301 Board
.56 in. (1.42 cm)
iSBC 301 Board + host board
Weight — .69 oz. (19 gm)

Electrical Characteristics
DC Power Requirements:
10 mA at +5 Volts incremental power

Environmental Characteristics
Operating Temperature — 0° to +55° C
Relative Humidity — to 90% (without condensation)

Reference Manuals
All necessary documentation for the iSBC 301 MULTIMODULE board is included in the CPU board Hardware Reference Manual (NOT SUPPLIED)

iSBC 80/24 — Order No. 142648-001
iSBC 88/40 — Order No. 124978-001

Manuals may be ordered from any Intel sales representative, distributor office, or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

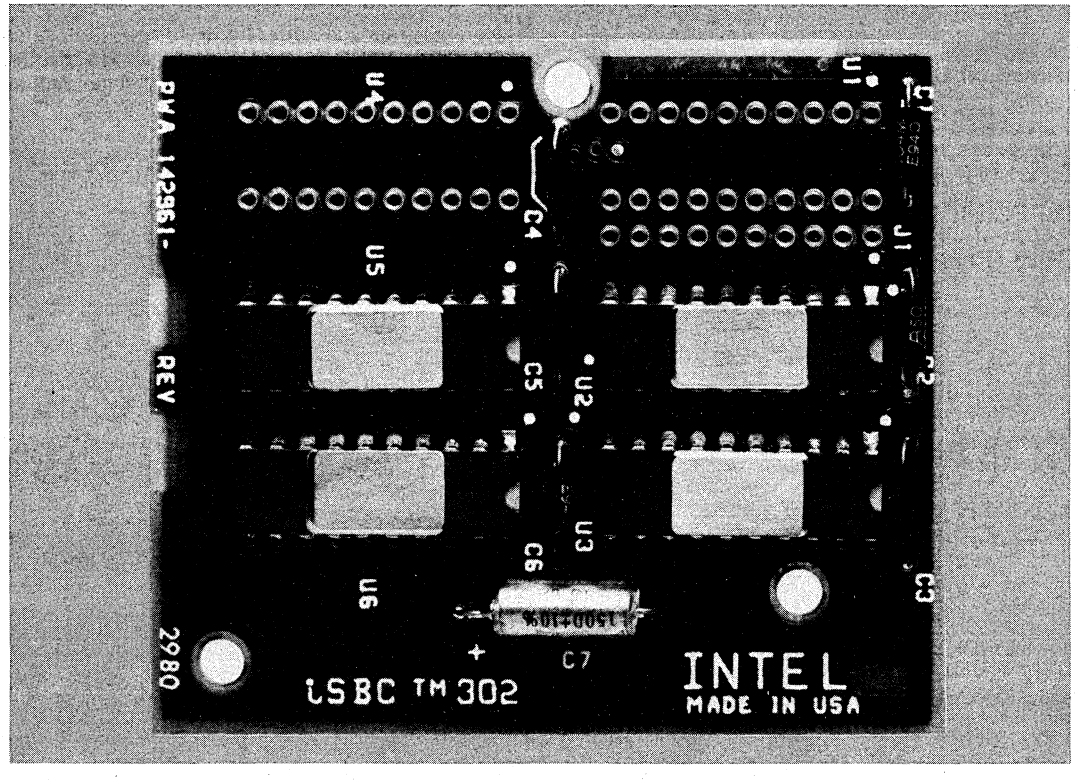
SPECIFICATIONS

Part Number	Description
SBC 301	4K Byte RAM MULTIMODULE Board

iSBC® 302
8K-BYTE MULTIMODULE™ RAM

- **Expands on-board memory of the iSBC® 86/05 and iSBC® 88/25 Single Board Computers**
- **Uses four Intel® 2168 static RAMs**
- **Single +5V supply**
- **On-board memory expansion eliminates system bus latency and increases system throughput**
- **Reliable mechanical and electrical interconnection**

The Intel iSBC 302 8K-Byte MULTIMODULE RAM provides simple, low-cost expansion to double the RAM capacity on the iSBC 86/05 Single Board Computer to 16K bytes or increase RAM capacity on the iSBC 88/25 Single Board Computer to 12K bytes. This offers system designers a new level of flexibility in implementing system memory. Because the MULTIMODULE memory is configured on-board, it can be accessed as quickly as the standard on-board iSBC 86/05 or iSBC 88/25 memory, eliminating the need for accessing additional memory via the MULTIBUS system bus. As a result, the iSBC 302 board provides a high-speed, cost-effective solution for systems requiring incremental RAM expansion.



FUNCTIONAL DESCRIPTION

The iSBC 302 board measures 2.60″ by 2.30″ and mounts above the RAM area on the iSBC 86/05 or iSBC 88/25 Single Board Computer. The iSBC 302 MULTIMODULE board contains four 4K x 4 static RAM devices and sockets for two of the RAM devices on the iSBC 86/05 board. With the iSBC 302 module mounted on the iSBC 88/25 board, the two sockets on the iSBC 302 module may be filled with 4K x 4 static RAMs. The two sockets on the iSBC 302 module have extended pins which mate with two sockets on the base board. Additional pins mate to the power supply and chip select lines to complete the electrical interface. The mechanical integrity of the assembly is assured with nylon hardware securing the module in two places. With the iSBC 86/05 or iSBC 88/25 board mounted in the top slot of an iSBC 604/614 cardcage, sufficient clearance exists for the mounted iSBC 302 option. If the iSBC 86/05 or iSBC 88/25 board is inserted into some other slot, the combination of boards will physically (but not electrically) occupy two cardcage slots.

SPECIFICATIONS

Word Size

8/16 bits

Memory Size

16,384 bytes of RAM

Cycle Time

Provides "no wait state" memory operations on the iSBC 86/05 board at 5 MHz or 8 MHz or the iSBC 88/25 board at 5 MHz.

5 MHz cycle time — 800 ns
8 MHz cycle time — 500 ns

Memory Addressing

Memory addressing for the iSBC 302 MULTIMODULE board is controlled by the host board via the address and chip select signal lines.

With the iSBC 86/05 board:

The 8K bytes of RAM on the iSBC 302 board occupy the 8K-byte address space immediately after that of the iSBC 86/05 board's 8K RAM (i.e., default configuration —

iSBC 86/05 board's RAM — $00000\text{-}01FFF_H$
iSBC 302 board's RAM — $02000\text{-}03FFF_H$).

With the iSBC 88/25 board:

The 8K bytes of RAM on the iSBC 302 board occupy the 8K byte address space immediately after that of the iSBC 88/25 board's 4K RAM (i.e., default configuration —

iSBC 88/25 board's RAM — $0\text{-}0FFF_H$
iSBC 302 board's RAM — $01000_H\text{-}02FFF_H$).

Physical Characteristics

WIDTH — 2.6 in. (6.60 cm)

LENGTH — 2.3 in. (5.84 cm)

HEIGHT — 0.56 in. (1.42 cm) iSBC 302 board + iSBC 86/05 or iSBC 88/25 board

WEIGHT — 1.25 oz (35 gm)

Electrical Characteristics

DC POWER REQUIREMENTS — 720 mA at +5V incremental power

Environmental Characteristics

OPERATING TEMPERATURE — 0°C to +55°C

RELATIVE HUMIDITY — to 90% (without condensation)

Reference Manuals

All necessary documentation for the iSBC 302 MULTIMODULE board is included in the CPU board Hardware Reference Manuals (NOT SUPPLIED).

iSBC 86/05 — Order No. 143153-001
iSBC 88/25 — Order No. 143825-001

Manuals may be ordered from any Intel sales representative, distributor office, or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

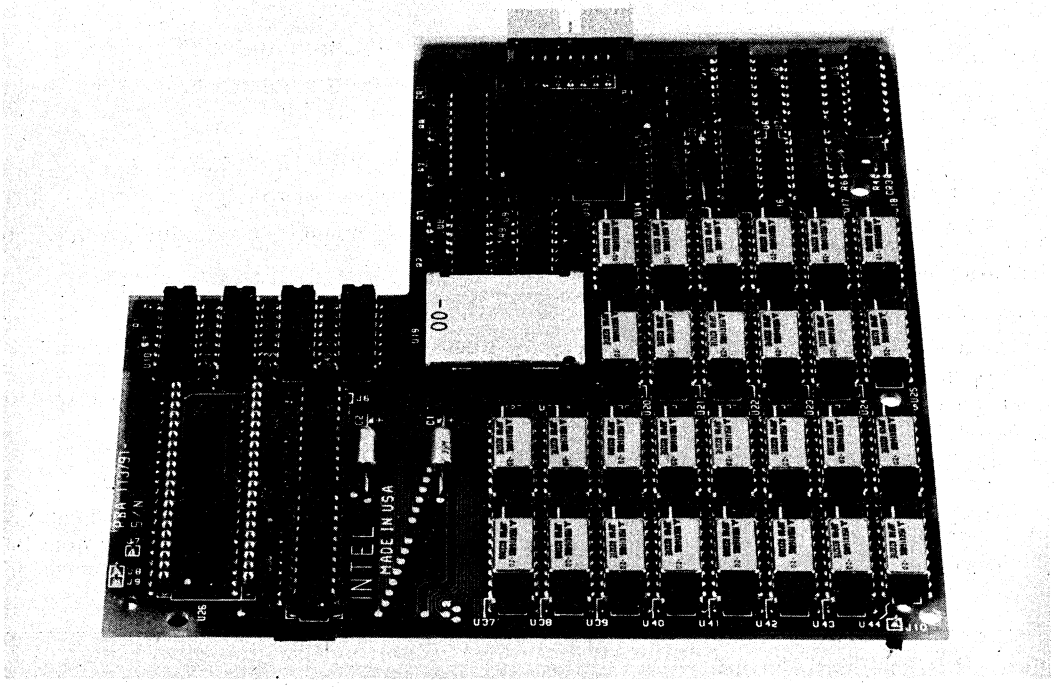
ORDERING INFORMATION

Part Number	Description
SBC 302	8K-Byte MULTIMODULE RAM

iSBC® 304C 128K Byte ECC MULTIMODULE™ Board

- **The iSBC® 304C RAM MULTIMODULE™ board provides 128K bytes of dual port RAM expansion for the iSBC® 86/30 board**
- **Single-bit error correction and double-bit error detection via Intel 8206 ECC device for all 256K bytes of on-board RAM**
- **Simple, reliable, mechanical and electrical interconnection**
- **On-board memory expansion for the iSBC® 86/30 Single Board Computer eliminates MULTIBUS® system bus latency and increases system throughput**
- **Low power requirements**

The iSBC® 304C RAM MULTIMODULE™ board provides simple, low-cost expansion of iSBC 86/30 board memory. The 128K bytes provided by the iSBC 304C board, together with the 128K bytes supplied on the iSBC 86/30 board, provide 256K bytes of dual-port, on-board RAM. The iSBC 304C board provides single-bit error correction and double-bit error detection for the full 256K bytes of on-board memory via the Intel 8206 error checking and correction (ECC) device. The MULTIMODULE board offers system designers a new level of flexibility in defining and implementing Intel single board computer systems. Because the iSBC 304C MULTIMODULE board expands the memory configuration on-board, the entire 256K bytes can be accessed quickly by eliminating the need to access additional memory via the MULTIBUS® system bus.



August, 1983
230762-001

FUNCTIONAL DESCRIPTION

General

The iSBC 304C MULTIMODULE board provides an incremental 128K bytes of on-board dual ported RAM to the iSBC 86/30 board memory. The iSBC 304C MULTIMODULE board also provides error checking and correction for all 256K bytes of on-board memory on the iSBC 86/30 board.

Error Checking and Correcting (ECC)

Error checking and correction is accomplished with the Intel 8206 ECC device.

The 8206 Error Checking and Correction Unit is a high-speed device that provides error detection and correction. During write cycles, the 8206 receives the write data and generates corresponding ECC check bits. During read cycles, the 8206 receives the read data and corresponding ECC check bits, corrects the data if necessary, and outputs the correct data.

During the write cycles, the ECC control logic coordinates full-word read/write (read-modify-write) operations between the 8203 Dynamic RAM Controller and the 8206 Error Checking and Correction Unit.

The data RAM array provides an additional 128K bytes of memory capacity for the iSBC 86/30 board. The check bit RAM array provides the ECC check bit storage space for both the 128K byte data RAM array on the iSBC 86/30 board and the 128K byte data RAM array on the iSBC 304C MULTIMODULE board.

The iSBC 304C MULTIMODULE board detects double-bit errors and corrects single-bit errors. No diagnostic modes or registers are present. If an error is detected by the 8206, the ERROR line is raised. If the error is a single-bit error, the error is corrected and the CE (correctable error) goes active. The error logic makes the CEINT (correctable error interrupt) line active. If the detected error is not a correctable error (a multiple bit error), the error logic activates the UCEINT (uncorrectable error interrupt) line to the iSBC 86/30 board.

Installation

Each MULTIMODULE board contains the 8206 error checking and correction (ECC) unit, ECC control logic,

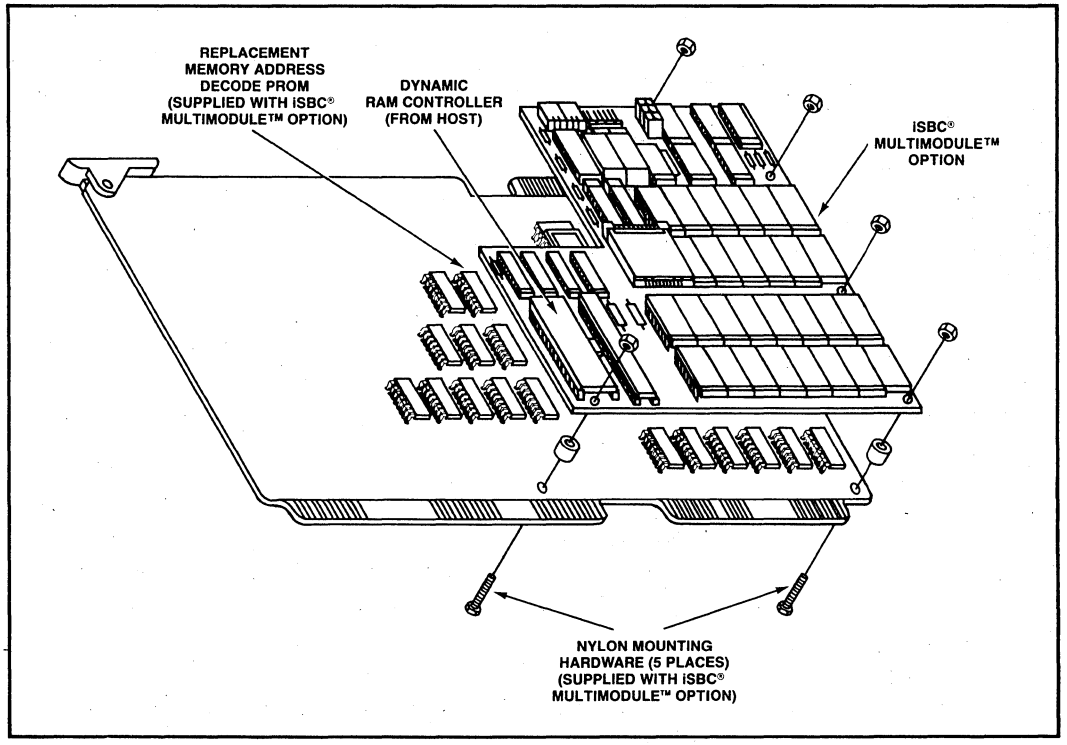


Figure 1. Installation of the iSBC® 304C ECC MULTIMODULE™ RAM on the iSBC® 86/30 Single Board Computer.

dynamic RAM devices and sockets for the Intel 8203 dynamic RAM controller. To install the module, the latches and controller from the host CPU board are removed. The 8203 is inserted into a socket on the RAM MULTIMODULE board. The module is then mounted onto the host board. Pins extending from the MULTIMODULE board mate with device sockets underneath (see Figure 1). Additional pins mate to supply other signals to complete the electrical interface.

The module is then secured at five additional points with nylon hardware to ensure the mechanical integrity of the assembly.

To complete the installation, one socketed PROM is replaced on the host CPU board with the one supplied with the MULTIMODULE board kit. This PROM is the MULTIBUS address decode PROM which allows the host board logic to recognize its expanded on-board memory compliment.

Note: The iSBC 304C board is larger than other MULTIMODULE boards. Systems configured using the iSBC 304C board are allowed only the following configurations:

— iSBC 304C + one single-wide iSBX™ MULTIMODULE
— iSBC 304C + one double-wide iSBX™ MULTIMODULE
— iSBC 304C + two single-wide iSBX™ MULTIMODULE

SPECIFICATIONS

Word Size

8 or 16 bits (16-bit data paths)

Memory Size

iSBC® 304C module — 128K bytes ECC RAM

The iSBC 304C MULTIMODULE board provides ECC capability for the additional 128K bytes of RAM on the iSBC 86/30 board.

Cycle Time

16-bit Read Cycle — 800 nsec
16-bit Write Cycle — 800 nsec
Byte Write Cycle — 1100 nsec

The addition of the ECC capability adds one wait state to on-board accesses and less than 100 nanoseconds to system accesses to the dual-port memory for 16-bit reads and writes.

Memory Addressing

CPU ACCESS

iSBC® 304C (with iSBC® 86/30) — 256K bytes (total capacity); 0-3FFFFH (address range)

MULTIBUS® System Bus Access

Jumper selectable for any 32K boundary, but not crossing a 256K boundary on the iSBC 86/30 host board.

Interface

The iSBC 304C MULTIMODULE board option is designed specifically for the iSBC 86/30 host board.

Private Memory Allocation

Segments of the combined host/MULTIMODULE RAM board memory may be configured as a private resource, protected from MULTIBUS system access. The amount of memory allocated as a private resource may be configured in increments of 25% of the total on-board memory ranging from 0% to 100%. The iSBC 304C module mounted on the iSBC 86/30 board, therefore, supports private allocation of 64K, 128K, 192K or 256K bytes of RAM memory.

Physical Characteristics

Width — 7.05 in. (17.9 cm)

Length — 12 in. (30.48 cm)

Height — 0.5 in. (1.2 cm)

iSBC 86/30 board + iSBC 304C board combined thickness — 1.1 in (2.8 cm)

(Height = height of the components above the printed circuit board.)

(Thickness = distance from component leads below board to maximum height of components above board.)

Electrical Characteristics

DC POWER REQUIREMENTS

(To be supplied by the iSBC 86/30 board)

Main Power (+5V) — Voltage +5VDC ±5%; Current 1.10 Amps

Battery Back-up (+5VB) — Voltage +5VDC ±5%; Current 0.70 Amps

Environmental Characteristics

Operating Temperature — 0°C to 55°C

Relative Humidity — to 90% (without condensation)

Reference Manual

All necessary documentation for the iSBC 304C MULTIMODULE board is contained in the iSBC 304C Hardware Reference Manual. Order Number: 122153-001

Additional information regarding the iSBC 86/30 single board computer can be found in the iSBC 86/30 board data sheet (Order Number 210219-001) and iSBC 86/30 Hardware Reference Manual (Order Number 144044-001).

Literature may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, CA 95051

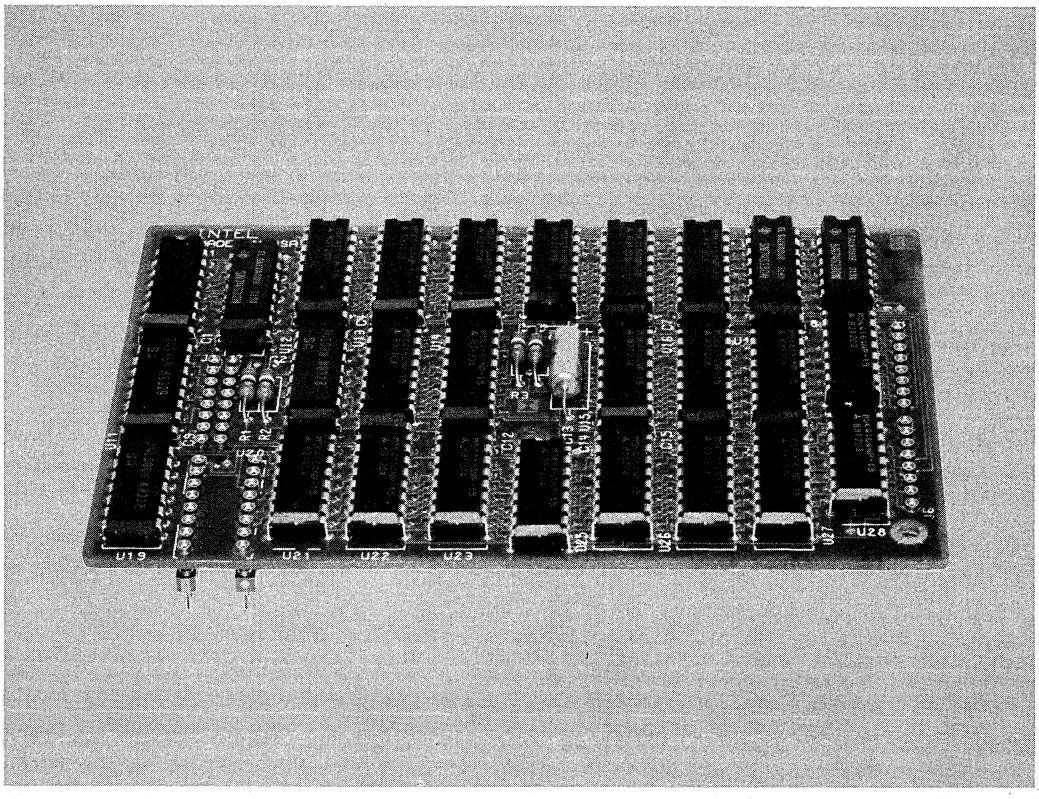
ORDERING INFORMATION

Part Number	Description
SBC 304C	128K ECC MULTIMODULE™ option for iSBC® 86/30 board

iSBC® 307 128K BYTE RAM MULTIMODULE™ BOARD WITH PARITY

- On-board memory expansion for the iSBC® 188/48 Advanced Communicating Computer
- Expands the iSBC® 188/48 on-board RAM to 192K Bytes and provides parity for total 192K Bytes
- Simple, reliable mechanical and electrical interconnection
- iSBC® 307 module provides 128K Bytes of RAM with parity expansion
- Low power requirements

The iSBC® 307 RAM MULTIMODULE board provides simple, low-cost expansion of the iSBC 188/48 board memory. The 128K Bytes of memory provided by the iSBC 307 MULTIMODULE board and the 64K Bytes of memory supplied by the iSBC 188/48 board provides a total of 192K Bytes of on-board RAM. The iSBC 307 MULTIMODULE board generates parity during all write operations and performs parity checking during all read cycles. This MULTIMODULE board offers systems designers flexibility in designing and implementing data communications networks.



OCTOBER 1984
ORDER NUMBER: 230970-002

FUNCTIONAL DESCRIPTION

The iSBC 307 MULTIMODULE board provides an additional 128K Bytes of on-board RAM to the iSBC 188/48 Advanced Communicating Computer. The iSBC 307 board also provides parity generation and check of all write and read cycles to the total 192K Bytes of on-board memory including the 64K Bytes on the iSBC 188/48 board.

PARITY GENERATION AND CHECKING

The iSBC 307 MULTIMODULE board generates parity during all write operations and performs parity checking during all read operations. When a parity error is detected, the iSBC 307 board generates a parity interrupt. Polarity can be set to even or odd through the Parity Test Pin. For upward capability with future 16-bit Single Board Computer boards, two parity generators are provided. The parity generators (74S20) generate parity for the selected bank of memory.

INSTALLATION:

The iSBC 307 MULTIMODULE board is mounted onto the host board. The board is secured at 1 point with nylon hardware to ensure the mechanical integrity of the assembly. The mounting technique used for the iSBC 188/48 is illustrated below in Figure 1. To install the module, the 2164 Dynamic RAM component from position U101 on the iSBC 188/48 board is removed. This 2164 is then installed into socket U20 on the iSBC 307 board.

The module is then secured by installing one of the supplied screws through the top of the hole above position U24 of the iSBC 307 board. Pins extending from the iSBC 307 MULTIMODULE board mate with connector receptacles located on the iSBC 188/48 board. When all the pins are properly mated, the MULTIMODULE board is seated to the iSBC 188/48 by pressing down firmly and evenly on the iSBC 307 board. Installing the final plastic nut onto the iSBC 307 board securing screw from the trace side of the iSBC 188/48 board completes installation.

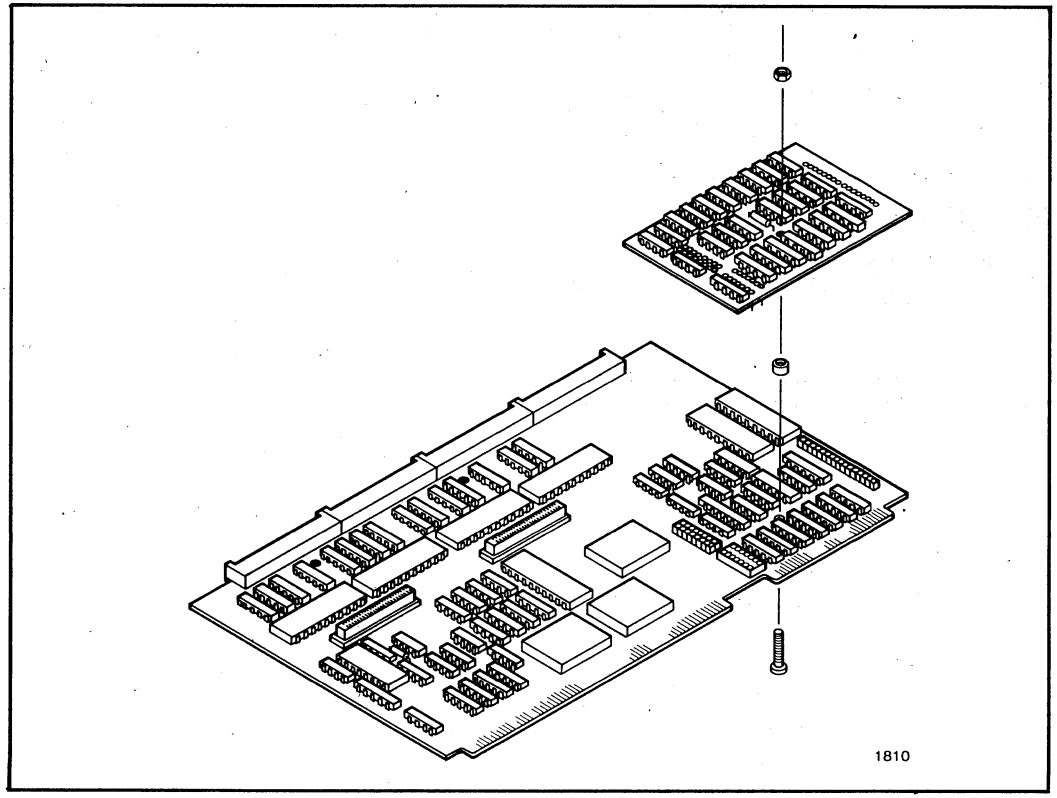


Figure 1. Installation of the iSBC® 307 RAM MULTIMODULE™ board on the iSBC® 188/48 Advanced Communicating Computer

SPECIFICATIONS

Word Size: 8 bits or 16 bits

Memory Size: 131,072 Bytes

Memory Access Time:

	MIN	MAX
On-Board (Read/Fetch)	0 wait state	1 wait state
On-Board (Write)	0 wait state	2 wait states
MULTIBUS Access (command to xack)	1108 nsec	1275 nsec

Refresh Timing:

Refresh Delay Time: 1002 ns. or 1 μsec. (Increase to normal access time due to a refresh cycle occurring)

Refresh Cycle Time: 1 Refresh cycle ever 15.6 μsec.

Address Selection:

Memory: Fixed by the baseboard.

Interface:

The iSBC 307 MULTIMODULE board option is specifically designed for the iSBC 188/48 Advanced Communication Computer board.

Physical Characteristics:

WEIGHT: 3.00 oz. (85.05 gm.)

WIDTH: 3.00 in. (7.62 cm.)

LENGTH: 4.45 in. (11.30 cm.)

HEIGHT: iSBC 188/48 board plus iSBC 307 MULTIMODULE board combined thickness is .724 inches max. (1.84 cm.) (As measured from the solder side of the iSBC 188/48 to the top of the components on the iSBC 307.)

Electrical Characteristics:

DC Power Requirements: Supplied by iSBC 188/48 board

Main Power (+5v): .44A (iSBC 307 stand-by)
.80A (iSBC 307 active)

(Active = memory being accessed either WRITE or READ)

ORDERING INFORMATION:

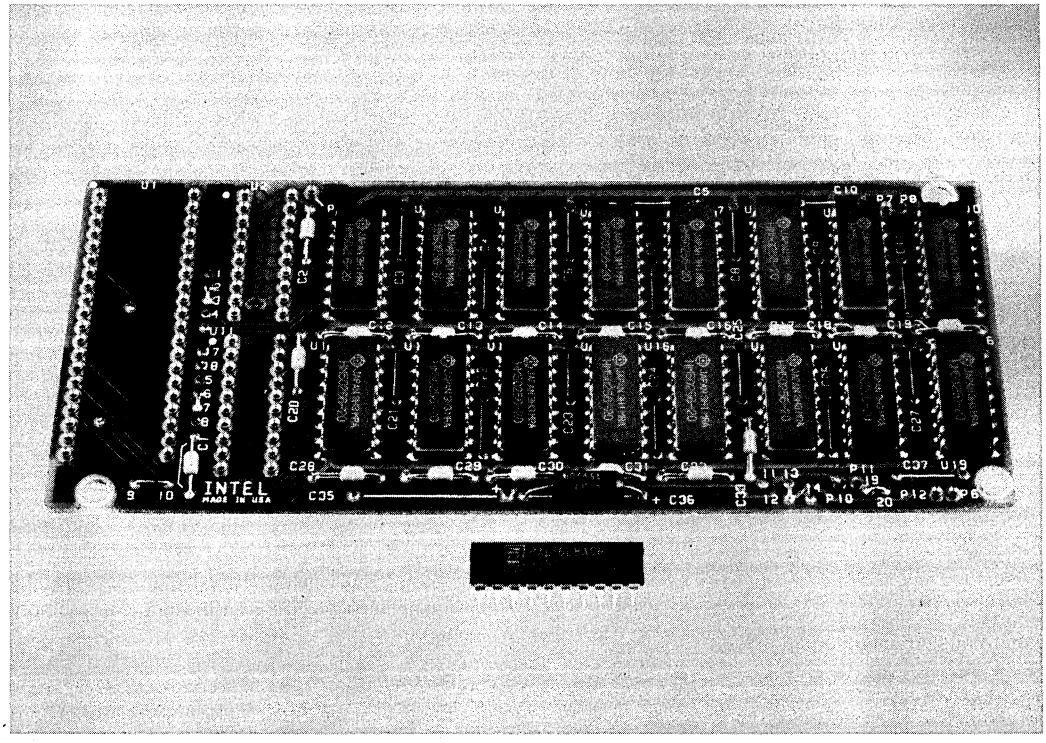
Part Number: Description
SBC 307 128K Byte RAM MULTIMODULE with Parity

Hardware Reference Manual: Covered in iSBC 188/48 Hardware Reference Manual Order Number: 146218-001

iSBC® 314
512K BYTE RAM MULTIMODULE™ BOARD

- **On-board memory expansion for the iSBC® 86/35 Single Board Computer**
- **iSBC® 314 module provides 512K bytes of dual port RAM expansion for the iSBC® 86/35 board**
- **Reliable mechanical and electrical interconnection**
- **Completes iSBC® 86/35 memory array providing a full megabyte page of system memory**
- **Increases system throughput by reducing accesses to MULTIBUS® global memory**
- **Low power requirements**
- **Battery backup capability**

The iSBC® 314 512K-Byte RAM MULTIMODULE™ board provides simple, low cost expansion to double the on-board RAM capacity of the iSBC 86/35 Single Board Computer host to one megabyte. This RAM MULTIMODULE option offers system designers a simple, practical solution to expanding and improving the memory capability and performance of the iSBC 86/35 board. The iSBC 314 memory is configured on-board and can be accessed as quickly as the standard iSBC 86/35 memory, eliminating the need for accessing additional memory via the MULTIBUS system bus.



OCTOBER 1984
ORDER NUMBER: 280000-002

FUNCTIONAL DESCRIPTION

The iSBC 314 MULTIMODULE board measures 2.40" by 5.75" and mounts above the RAM array on the iSBC 86/35 Single Board Computer. The iSBC 314 board contains sixteen 256Kbit x 1 dynamic RAM devices and three sockets; two for the memory latches and one for the Intel 8203 dynamic RAM controller. The addition of the iSBC 314 memory MULTIMODULE board to the iSBC 86/35 board makes possible a one megabyte single board solution; the full direct addressing capability of the iAPX 86 CPU.

To install the module, the latches and controller from the host iSBC 86/35 board, are removed and inserted into sockets on the iSBC 314 board. The module is then mounted onto the host board. Pins extending from the controller and latch sockets mate with device sockets underneath (see Figure 1). Additional pins mate to supply other signals to complete the electrical interface. The module is then secured at three additional points with nylon hardware to ensure the mechanical security of the assembly.

To complete the installation, one socketed PAL is replaced on the iSBC 86/35 board with the one supplied with the MULTIMODULE kit. This is the PAL which allows the host board logic to recognize its expanded on-board memory compliment.

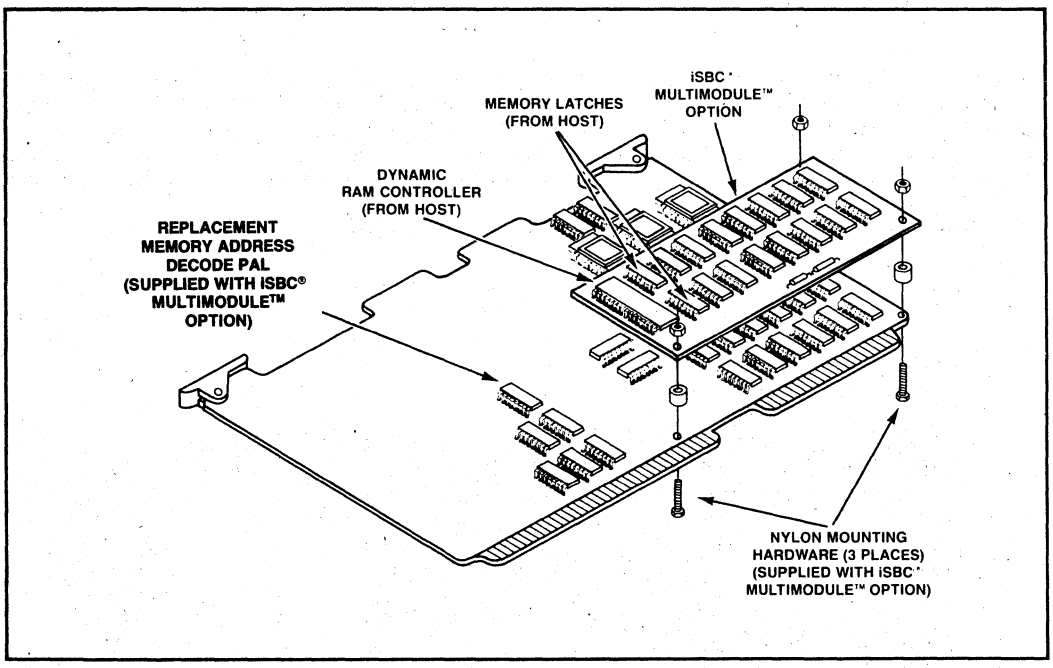


Figure 1. Installation of the MULTIMODULE™ RAM Module on the Host Single Board Computer

SPECIFICATIONS

Word Size

8 or 16 bits (16-bit data paths)

Memory Size

512K bytes RAM

System Cycle Time (8 MHz, 2 Wait States)

750 nsec (read); 750 nsec (write)

Memory Addressing

iSBC 314 module with iSBC 86/35 board — 1M bytes (total capacity); 0 — FFFFFH. (See Figure 2., Memory Allocation)

Interface

The interface for the iSBC 314 MULTIMODULE board option is designed only for the iSBC 86/35 host board.

Wait-State Performance

A significant performance advantage of 2 wait-states is achieved when accessing memory on-board the

iSBC 86/35 versus the performance of 6 wait-states when accessing memory off-board over the MULTIBUS. The iSBC 314 puts an additional 512K bytes of system memory on-board the iSBC 86/35 reducing execution time by as much as 70%.

Memory Allocation

Segments of the combined host/MULTIMODULE RAM may be configured to be accessed either from off-board or on-board resources. The amount of memory allocated as either public or private resource may be configured in a variety of sizes. The address range boundaries for the 1 megabyte RAM array of the iSBC 314 and iSBC 86/35 board combination are shown in Fig. 2 for accesses from both on-board and off-board resources.

Auxilliary Power

The low power memory protection option included on the iSBC 86/35 board supports the iSBC 314 module.

Physical Characteristics

Width — 2.4 in. (6.10 cm)

Length — 5.75 in. (14.61 cm)

Depth* — .72 in. (1.83 cm)

Weight — .13 oz. (59g)

***Note:** Combined depth including host board.

Electrical Characteristics

DC Power Requirements*

*Additional power required by the iSBC 314 MULTIMODULE is:

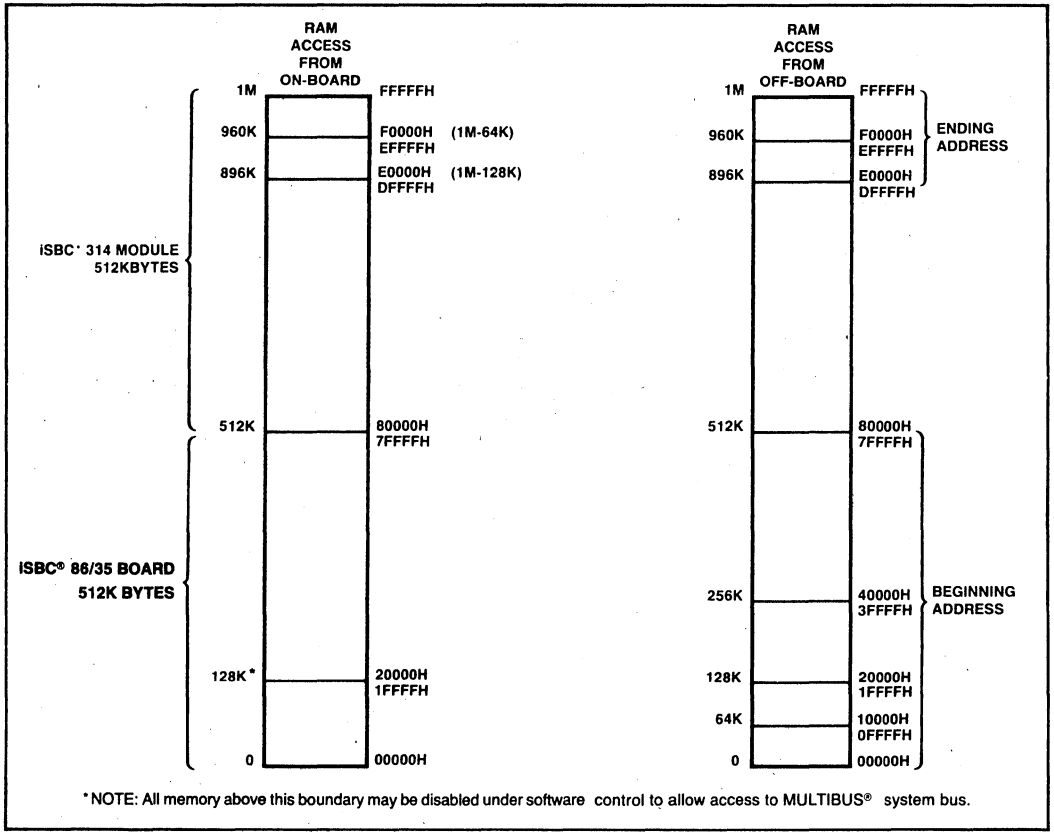
Typical: 60 mA @ +5 volts

Maximum: 140 mA @ +5 volts

Environmental Characteristics

Operating Temperature — 0°C to 55°C

Relative Humidity — to 90% (without condensation)



*NOTE: All memory above this boundary may be disabled under software control to allow access to MULTIBUS® system bus.

Figure 2. Address Range Selection

Reference Manual

All necessary documentation for the iSBC 314 MULTIMODULE board is included in the iSBC 86/35 Hardware Reference Manual (NOT SUPPLIED); Order Number: 146245-001.

Manuals may be ordered form any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, CA 95051.

ORDERING INFORMATION

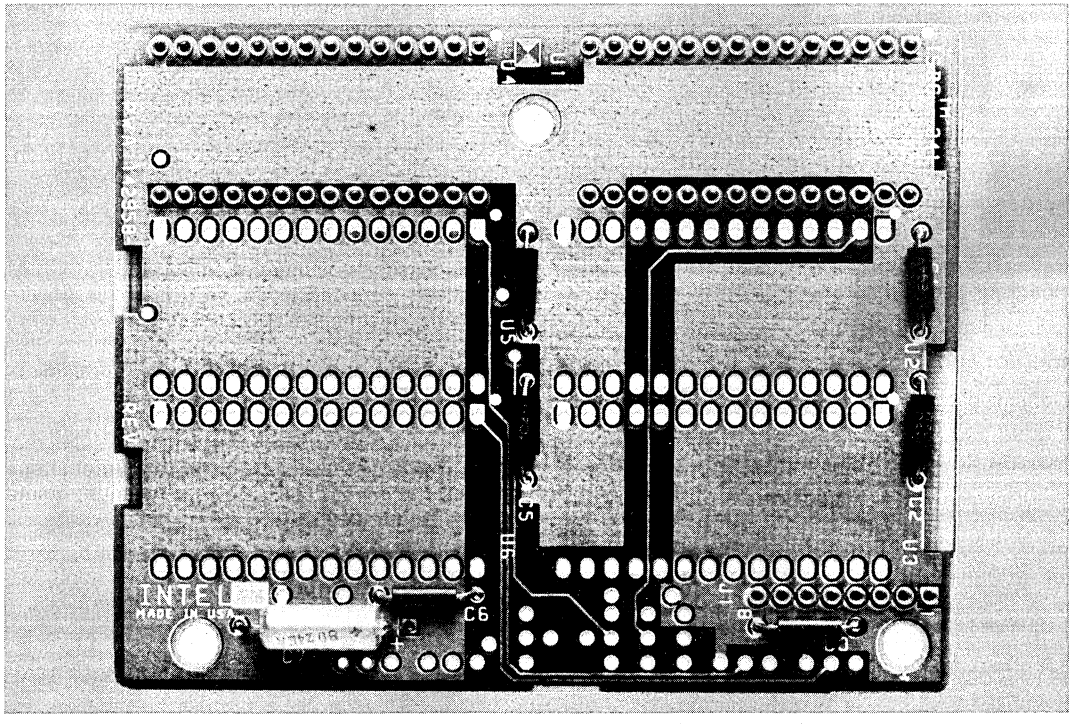
Part Number	Description
iSBC® 314	512K byte Memory MULTIMODULE™ option for iSBC 86/35 board

iSBC® 341
28-PIN MULTIMODULE EPROM

- **On-board memory expansion for iSBC® 86/05, iSBC®88/25, and iSBC®88/40 microcomputers**
- **Supports JEDEC 24/28-pin standard memory devices, including EPROMs, byte-wide RAMs, and E²PROMs**
- **Sockets for up to 64K bytes of expansion with Intel® 27128 EPROMs**
- **On-board expansion provides "no wait state" memory access with selected devices**
- **Simple, reliable mechanical and electrical interface**

The iSBC 341 28-pin MULTIMODULE EPROM board provides simple, low-cost expansion of the on-board EPROM capacity of the iSBC 86/05 Single Board Computer, the iSBC 88/25 Single Board Computer and the iSBC 88/40 Measurement and Control Computer. Four additional 28-pin sockets support JEDEC 24/28-pin standard devices, including EPROMs, byte-wide static and pseudo-static RAMs.

The MULTIMODULE expansion concept provides the optimum mechanism for incremental memory expansion. Mounting directly on the microcomputer, the benefits include low cost, no additional power requirements beyond the memory devices, and higher performance than MULTIBUS-based memory expansion.



AFN-01825A

FUNCTIONAL DESCRIPTION

The iSBC 341 28-pin MULTIMODULE EPROM option effectively doubles the number of sockets available for EPROM on the base microcomputer board on which it is mounted. The iSBC 341 board contains six 28-pin sockets. Two of the sockets have extended pins which mate with two of the sockets on the base board. Two of the EPROMs which would have been inserted in the base board are then reinserted in the iSBC 341 sockets. Additional interface pins also connect chip select lines and power. The mechanical integrity of the assembly is assured with nylon hardware securing the unit in two places.

Through its unique interface, the iSBC 341 board can support 8 or 16-bit data paths. The data path width is determined by the base board — being 8 bits for the iSBC 88/40 and iSBC 88/25 microcomputers, and 8/16 bits for the iSBC 86/05 board.

SPECIFICATIONS

Word Size

8 or 8/16 bits (determined by data path width of base board).

Memory Size

32K bytes with available technology (JEDEC standard defines device pin-out to 128K-bit devices).

Device Size (Bytes)	EPROM Type	Max. iSBC 341 Capacity (Bytes)
2K x 8	2716	8K
4K x 8	2732A	16K
8K x 8	2764	32K
16K x 8	27128	64K

Access Time

Varies according to base board and memory device access time. Consult data sheet of base board for details.

Memory Addressing

Consult data sheet of base board for addressing data.

POWER REQUIREMENTS

Devices[1]	Max. Current @ 5V ±5%
2716	420 mA
2732A	600 mA
2764	600 mA

NOTE:
1. Incremental power drawn from host board for four additional devices.

Auxiliary Power

There are no provisions for auxiliary power (battery backup) on the iSBC 341 option.

Physical Characteristics

WIDTH — 3.4 in. (8.64 cm)

LENGTH — 2.7 in. (6.86 cm)

HEIGHT — 0.78 in. (1.98 cm))*

WEIGHT — 5 oz (141.5 gm)

*Includes height of mounted memory devices and base board.

All necessary mounting hardware (nylon screws, spacers, nuts) is supplied with each kit.

Environmental Characteristics

OPERATING TEMPERATURE — 0°C to +55°C

RELATIVE HUMIDITY — to 90% (without condensation)

Reference Manuals

All necessary documentation for the iSBC 341 module is included in the CPU board Hardware Reference Manuals (NOT SUPPLIED)

iSBC 86/05 — Order No. 143153-001
iSBC 88/25 — Order No. 143825-001
iSBC 88/40 — Order No. 124978-001

Manuals may be ordered from any Intel sales representative, distributor office, or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

ORDERING INFORMATION

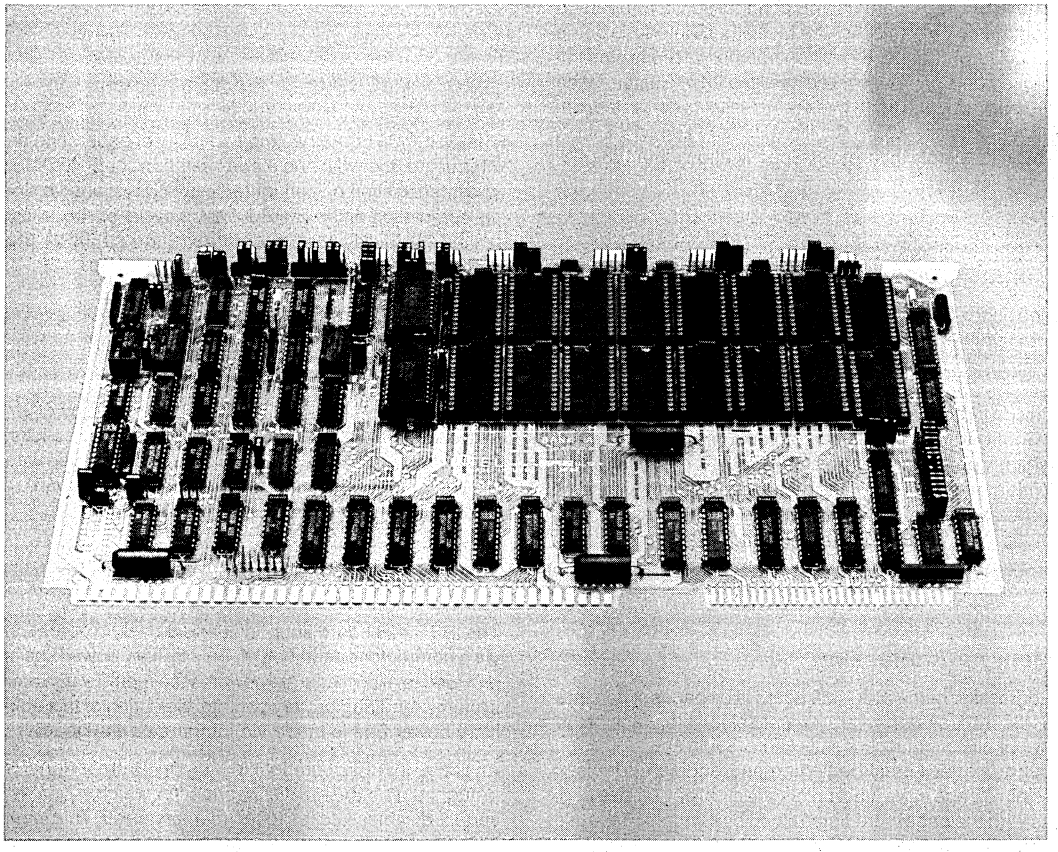
Part Number	Description
SBC 341	28-Pin MULTIMODULE EPROM

PRELIMINARY

iSBC® 428 UNIVERSAL SITE MEMORY EXPANSION BOARD

- **Supports EPROM, ROM, E²PROM, SRAM, IRAM and NVRAM**
- **iLBX™ BUS or MULTIBUS® Selectable**
- **Provides support for Battery Backup/ Memory Protect**
- **Sixteen 28 pin Universal sites**
- **Assignable anywhere within a 16 megabyte address space on 256K byte boundaries**
- **Jumper selectable base address on 4K byte boundaries**

The iSBC® 428 Universal Site Board is a member of Intel's complete line of Memory and I/O Expansion boards. The iSBC 428 Universal Site Memory Expansion Board interfaces directly to the iSBC 80, iSBC 88, or iSBC 86 Single Board Computers via the MULTIBUS® System Bus to expand system memory requirements, while system memory expansion requirements for iSBC 286 Single Board Computer can interface via either the MULTIBUS or the high speed iLBX™ Bus.



FUNCTIONAL DESCRIPTION

General

The iSBC 428 board contains sixteen 28 pin sockets. The actual capacity of the board is determined by the type and quantity of components installed by the user. The iSBC 428 board is compatible with five different types and densities of devices: the 2K by 8 thru 64K by 8 EPROM/ROM devices, 2K by 8 thru 8K by 8 "Five Volt Only, Enhanced" E²PROM devices, 512 by 8 thru 16K by 8 NVRAM (Non-Volatile RAM) devices, 2K by 8 thru 32K by 8 SRAM devices, and 8K by 8 IRAM (Integrated RAM) devices. In addition the board can be accessed by either the MULTIBUS System Bus or Intel's new high speed iLBX Bus.

iLBX™ Bus

The iSBC 428 board can be configured via jumpers to communicate with either the MULTIBUS interface or the iLBX Bus interface. Significant memory access time improvements can be realized over the iLBX Bus interface (versus the MULTIBUS interface) due to its dedicated, unarbitrated architecture. Additional information on the iLBX Bus is available in the iLBX Specification #144456-003.

Memory Banks

The sixteen sites on the iSBC 428 board are partitioned into two banks of 8 sites each. Within each bank the 8 sites are further partitioned into 2 groups of 4 sites each. Each group of 4 sites is configurable to each of the six device types described above via a "Configurator". The "Configurator" is an arrangement of push-on jumpers which configures each of the four groups of 4 sites. Within each bank devices of the same density must reside and within each group devices of the same type must reside (i.e. SRAM or EPROM).

Memory Addressing

Addressing of the iSBC 428 board is by pages. There are 64-256K pages which are jumper selectable. Each of the two banks are independently addressable and can reside in any page. Actual beginning and ending addresses within a page are a function of the actual device size and, as with the pages, are determined by jumpers. Because of the paging based memory addressing architecture more than one iSBC 428 board can be placed in a system.

Mode of Operation

The iSBC 428 board can operate in one of two modes: the 8 bit only mode or the 8/16 bit mode. The 8 bit mode provides the most efficient memory configuration for systems handling 8 bit data only. The 8/16 bit mode allows the iSBC 428 board to be compatible with systems employing 8 bit and 16 bit masters. The mode of operation is selected by on board jumpers and is available for both MULTIBUS and iLBX Bus configurations.

Memory Access

The iSBC 428 board has jumper selectable access times which allows the board to be tailored to the performance of the particular devices which are installed in the iSBC 428 board. The board can be configured via jumpers to accept devices with an access time range of 50 ns to 500 ns with a granularity of 50 ns and results in a board access time from 225 ns to 775 ns.

Interrupt

The iSBC 428 board has the capability of generating an interrupt for the write and erase operations of E²PROMS. The interrupt can be configured in two ways: one, to signal completion of the E²PROM write cycle, or two, allow polling by the system to determine the status of the E²PROM during the write programming time.

Inhibits

Inhibits are provided on the iSBC 428 board to allow ROM to overlay RAM for bootstrapping or diagnostic operations. Each bank of the iSBC 428 board can be overlayed with the system RAM by jumpers provided on the board.

Battery Backup

The iSBC 428 board supports battery backup operation via a connector on the board. An auxiliary power bus is provided to allow separate power to the memory array for systems requiring battery backup. Selection of this auxiliary power bus is made via jumpers on the board.

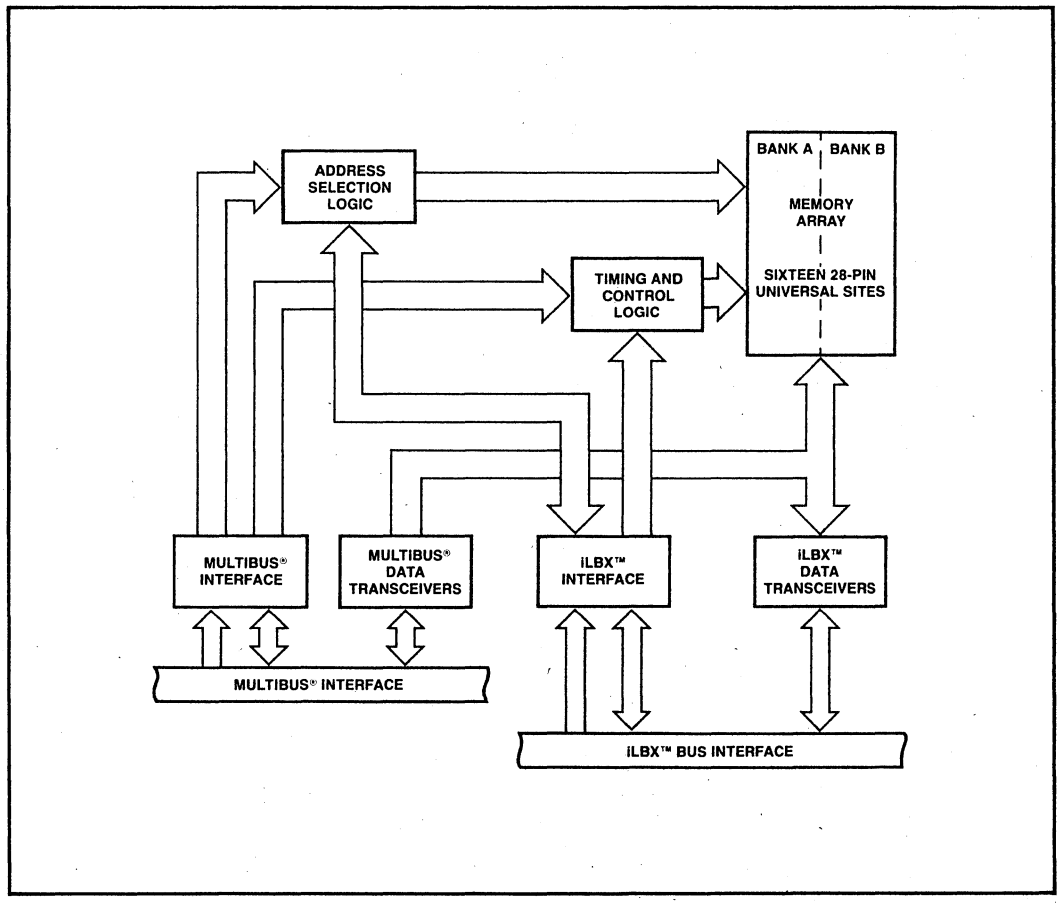
An active-low TTL compatible Memory Protect signal is brought out on the auxiliary connector which, when asserted, disables access to the memory array. This input is provided for the protection of Memory contents during system power-down sequences.

Devices Supported

Listed below are the current and future devices supported by the iSBC 428 board.

	Size							
Type	**512 x 8**	**2K x 8**	**4K x 8**	**8K x 8**	**16K x 8**	**32K x 8**	**64K x 8**	**Comments**
EPROM	—	2716	2732A	2764	27128	27256	27512	—
ROM	—	X	X	X	X	X	X	—
EEPROM	—	2817A	X	X	X	X	—	5V, Enhanced
SRAM	—	X	X	X	X	X	—	NMOS & CMOS
NVRAM	—	X	X	X	—	—	—	—
IRAM	—	—	—	2186	—	X	—	—

X-Denotes that the iSBC 428 board will support the device indicated but that it is not currently available from Intel.



iSBC® 428 Block Diagram

SPECIFICATIONS

Word Size

8 or 8/16 bits

Memory Size

Sockets are provided for up-to sixteen 28 pin devices which can provide up to 512K bytes of EPROM/ROM/SRAM.

Access Time

Jumperable from 225 to 775 ns with a granularity of 50 ns and is equivalent for both MULTIBUS and the iLBX Bus.

Power Requirements

$V_{CC} = 5$ volts $\pm 5\%$

$I_{CC} = 2.0$ amps, maximum, without any memory devices in the board.

Physical Characteristics

Length — 30.48 cm (12 inches)

Width — 17.15 cm (7.05 inches)

Depth — 1.27 cm (0.5 inches)

Environment

Operating Temperature — 0°C to +55°C

Relative Humidity — 90% non-condensing

Reference Manual

145696-001 — iSBC 428 Hardware Reference Manual (NOT SUPPLIED)

Additional Literature

9800683-04 — MULTIBUS Specification

144456-001 — The iLBX Specification

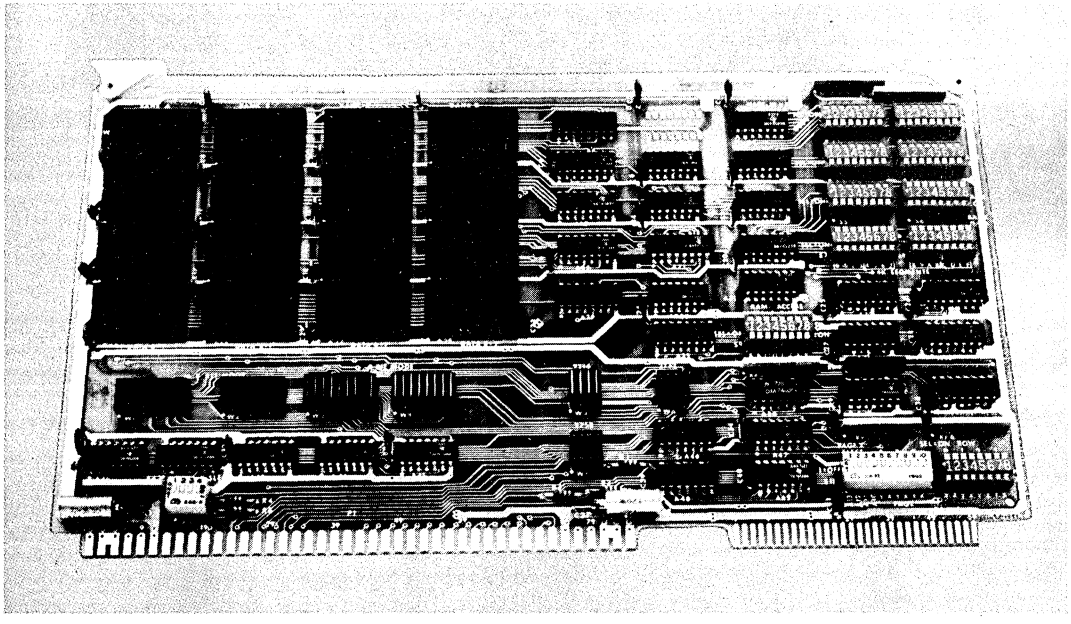
ORDERING INFORMATION

Part Number	Description
SBC 428	Universal Site Memory Expansion Board

iSBC® 464
64K BYTE EPROM EXPANSION BOARD

- **Provides EPROM/ROM expansion of iSBC® 80, iSBC® 86 and iSBC® 88 systems via direct MULTIBUS® interface**
- **Sockets for up to 64K bytes of EPROM**
- **Compatible with Intel® 2758, 2716 or 2732/2732A erasable PROMs**
- **Switch selectable base address on 4K byte boundaries for each memory bank**
- **Assignable anywhere within a 1 megabyte address space**
- **EPROM components which are not enabled are placed in standby power mode**
- **Requires a single +5V power supply**

The iSBC 464 is a member of Intel's complete line of iSBC memory and I/O expansion boards. The iSBC 464 board interfaces directly to the iSBC 80, iSBC 86 or iSBC 88 single board computers via the MULTIBUS system bus, to expand system EPROM memory capacity.



FUNCTIONAL DESCRIPTION

Memory Configuration

The iSBC 464 board contains sixteen sockets which provide a maximum of 64K bytes of memory expansion. The actual capacity of the board is determined by the type and quantity of EPROM components installed by the user. The board is compatible with three different sizes of Intel EPROM devices. These are the 1K byte 2758 EPROM, the 2K byte 2716 EPROM, and the 4K byte 2732 EPROM.

Mode of Operation — The iSBC 464 board can operate in one of two modes: the 8 bit only mode or the 16/8 bit mode. The 8 bit mode provides the most efficient memory configuration for systems handling 8 bit data. The 16/8 bit mode allows 16 bit words to be accessed by 16 bit processors. In the 16/8 bit mode, 16 bit and 8 bit microprocessors may also access either the high order byte or the low order byte of a 16 bit word. The mode of operation is selected by placing two option jumper blocks in the appropriate sockets.

Memory Banks — When used in the 8 bit mode, the iSBC 464 board is organized into four banks (labeled A-D) of four sockets each. Depending on the type of memory components used, each bank may contain a maximum of 4K, 8K or 16K bytes of memory. Unused memory sockets may be deselected by bank or individually in bank D. Deselecting a bank or individual socket frees that address space for use elsewhere in the system. In the 16/8 bit mode, banks A & B and C & D are paired together to form two banks (labeled AB, CD) which are 16 bits wide. Each of these banks has four socket pairs. Bank AB may be deselected as a single unit. Socket pairs in bank CD may be deselected individually. Thus, board configurations using fewer than 16 memory components do not fill memory address space with unused sockets. Selection/deselection is accomplished by setting switches on the board.

Memory Access Time — The iSBC 464 board operates with one of 15 switch selectable memory access times ranging from 35 to 1435 nanoseconds. This feature allows the board to be tailored to the performance of the installed components and the system CPU.

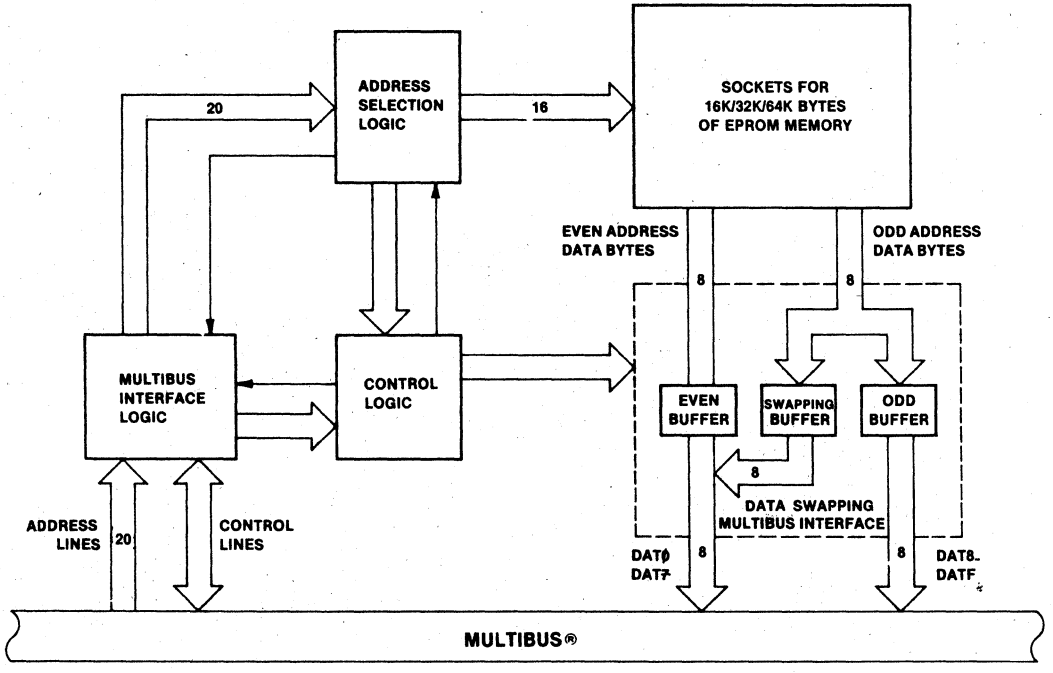


Figure 1. iSBC® 464 Block Diagram

Memory Addresses

Switch selectable options on the iSBC 464 board allow the board to be assigned anywhere within a 1 megabyte address space. In either operating mode, the base address of each memory bank may be set to any 4K byte boundary within a 64K byte memory page. There is one exception. If the 4K byte devices are used in the 16/8 bit mode, then base addresses are restricted to 8K byte boundaries. If the board is used in a system with an address range greater than 64K bytes, memory on the iSBC 464 board may reside in one or two 64K byte memory pages. Any two pages out of a possible 16 may be chosen by setting switches on the board.

Standby Power Operation

The iSBC 464 board takes advantage of the standby modes of the Intel 2758, 2716 and 2732. When they are not enabled, these components draw as little as 25% of their active level power with no degradation in access time. The iSBC 464 board is designed so that only two memory components are enabled during a read operation.

RAM Overlap

Memory banks of the iSBC 464 board can be overlapped with the addresses of system RAM by setting on-board switches. The process of addressing a memory bank will drive the Inhibit RAM (INH1/) signal true. This signal is issued to the MULTIBUS system bus in order to prevent any MULTIBUS accessable RAM in the system from responding to the current address. If an EPROM is addressed which has its corresponding RAM overlap switch on, an access time of 15 clock cycles is imposed. This allows overlapped dynamic RAM to refresh before the address on the MULTIBUS is changed. The RAM overlap feature does not apply to RAM which is not on the MULTIBUS system bus.

SPECIFICATIONS

Word Size

8 bits or 16 and 8 bits

Memory Size

Sockets are provided for up to 16K bytes in 1K increments or 32K bytes in 2K increments or 64K bytes in 4K increments

Compatible Intel® Memory

EPROM — 2758 or 2716 or 2732

INTERFACE — All 20 address, 16 data, and 6 control signals are TTL compatible and Intel MULTIBUS compatible

Electrical Characteristics

DC Power (max)

V_{CC}: +5V DC ±5%
I_{CC}: 1.1 amps without EPROMs
I_{CC}: 1.6 amps with (16) 2716s or 2758s
I_{CC}: 1.3 amps with (16) 2732s or 2732As

Connectors

Bus — 86-pin double-sided PC edge connector with 0.40 cm (0.156 in.) contact centers

Mating Connector — Viking 3KH43/9AMK12 or compatible connector

Physical Characteristics

Length — 30.48 cm (12 in.)
Height — 17.15 cm (6.75 in.)
Depth — 1.27 cm (0.5 in.)
Weight — 294 gm (10.5 oz) without EPROM

Environment

Operating Temperature — 0°C to +55°C
Relative Humidity Limits — <90% non-condensing

Reference Manual

9800643A — iSBC 464 Memory Expansion Board Hardware Reference Manual (NOT SUPPLIED)

Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

ORDERING INFORMATION

Part Number	Description
SBC 464	64K EPROM Expansion Board

PRELIMINARY

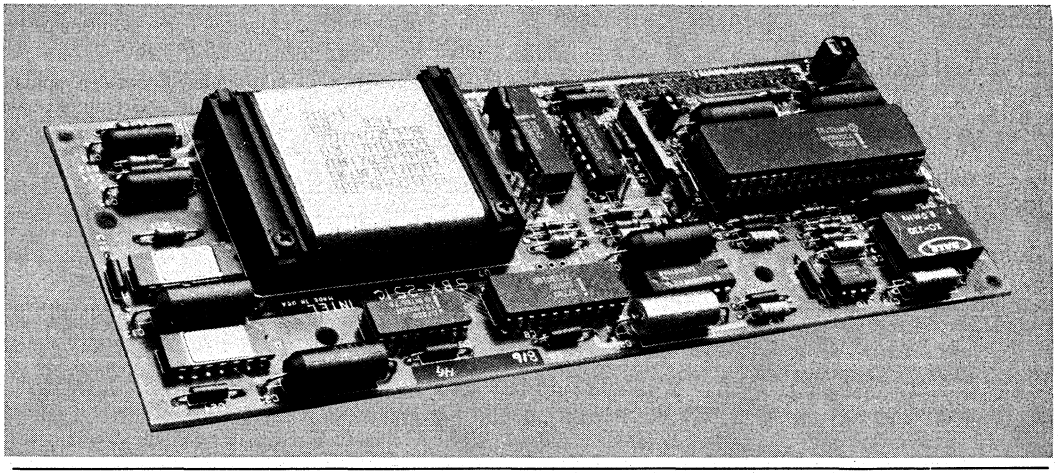
iSBX™ 251 and 251C BUBBLE MEMORY MULTIMODULE™ BOARD

- iSBX™ 251 0-60°C
  iSBX™ 251C 10-40°C
- iSBX™ MULTIMODULE™ Bus Compatible
- Capacity: 128K Bytes Bubble Memory Storage
- Performance:
  —Average Access Time: 48ms
  —Burst Data Rate: Up to 50K Bytes/Sec
- Compatibility with Host DMA Controller
- Non-Volatile Storage
- High Reliability Under Harsh Environments
- Fast Access Storage Option on iPDS™ System
- Automatic Error Correction
- Operates from Standard +5V and +12V Power Supplies
- Power Fail Data Protection
- Low Power Consumption

The Intel iSBX 251 and iSBX 251C bubble memory MULTIMODULE boards are completely assembled and tested Non-Volatile 128K-byte memory boards based on the Intel 7110 one-megabit bubble memory and support chips. The iSBX 251 and iSBX 251C boards are Intel's easiest to use bubble solutions. The iSBX 251 and iSBX 251C MULTIMODULE boards may be designed into Intel SBC products with iSBX connectors as well as into any user manufactured microprocessor board. The bubble memory support circuitry and SBX connector on the iSBX 251 and iSBX 251C boards provide the user with a simple interface to the bubble memory.

The iSBX 251 and iSBX 251C boards are featured as an option on the new Intel Personal Development System as a fast access storage option designed to emulate disk. Typically, the bubble memory option provides a 2X improvement in system performance when compared with a floppy disk. Use of the iSBX 251 or iSBX 251C board with the iPDS system enhances iPDS™ system portability, performance and reliability.

The iSBX 251 and iSBX 251C boards differ in specified operating temperature ranges. The iSBX 251 board operating temperature is 0-60°C. The iSBX 251C boards operating temperature is 10-40°C. These boards plug into any Intel iSBX single board computer or other processor board with an iSBX connector. The iSBX 251 board meets Intel iSBX specifications.



FUNCTIONAL DESCRIPTION

The Intel iSBX 251 and iSBX 251C bubble memory MULTIMODULE boards are completely assembled and tested non-volatile memory boards. They consist of the Intel 7110 bubble memory and support circuitry mounted on a double-wide MULTIMODULE board. The bubble memory support circuitry includes the Intel 7220-1 Bubble Memory Controller (BMC) through which the host processor communicates with the bubble storage. See Figure 1 for a system and bubble memory block diagram.

The BMC provides a convenient 8 bit bidirectional bus that requires only two port or I/O addresses. One port is used to transfer data while the other is used to send commands or view operational status. A set of sixteen commands are available to initiate and monitor a bubble memory data transfer. (Refer to the 7220-1 data sheet for more detailed information on the BMC commands).

The iSBX 251 provides the designer with three I/O modes of data transfer for complete flexibility. I/O mode selection is accomplished through the use of on-board jumpers and user software:

1. Polled
2. Interrupt-driven mode
3. Direct Memory Access (DMA) mode

DMA mode requires the use of a DMA controller on the host board.

The iSBX 251 and iSBX 251C boards plug into any Intel iSBC® Single Board Computer or processor board with an iSBX connector. This arrangement frees the MULTIBUS for other traffic while the host iSBC board accesses the bubble memory.

OPERATION

Like many high density peripheral storage devices, bubble memory data is organized serially in pages rather than bytes. Data transfers are accomplished under software control by the 7220-1 BMC. The 7220-1 partitions the one megabit bubble memory into 2048 pages of either 64 or 68 bytes in length. The page length is dependent upon the use of error detection and correction—64 bytes with error correction and 68 bytes without. Data transfers are specified in terms of whole pages. Therefore the minimum amount of data that can be transferred during one read or write command is 64 or 68 bytes. Automatic error correction may be selected by enabling a flag in the 7220-1 BMC.

The iSBX 251 board can be configured to operate in polled mode, interrupt mode, or DMA mode. In the polled mode, the host processor periodically reads the 7220-1 BMC status register to obtain information about completion or termination of commands, error conditions, and the BMC's readiness to transfer data or accept a new command.

In the interrupt-driven mode, an interrupt is issued by the 7220-1 BMC when its internal buffer is ready to accept 22 bytes of data during a write operation. In a read operation, an interrupt is issued whenever 22 bytes of data are available for reading by the host processor in addition to data transfers. The BMC will also issue an interrupt to indicate the completion of a command or the presence of error conditions.

With the assistance of a direct memory access controller on the board hosting the iSBX 251, the BMC can transfer large blocks of data with a single I/O request. DMA mode makes use of the BMC's handshaking ability with a DMA controller.

Regardless of the mode of data transfer, the host processor or DMA controller must be capable of maintaining a data rate of 12.5K bytes/sec.

MOUNTING TECHNIQUE

As shown in Figure 2, the iSBX 251 board plugs into a host board via the iSBX connector and is secured by three spacers with screws. A double-wide iSBX MULTIMODULE board is used and two MULTIBUS card slots are occupied in addition to the card slot for the base board. Dimensions of the board are given in Figures 3 and 4. Although the iSBX 251 board male connector has the standard 36 pins, this board also plugs into the expanded 44-pin female connector.

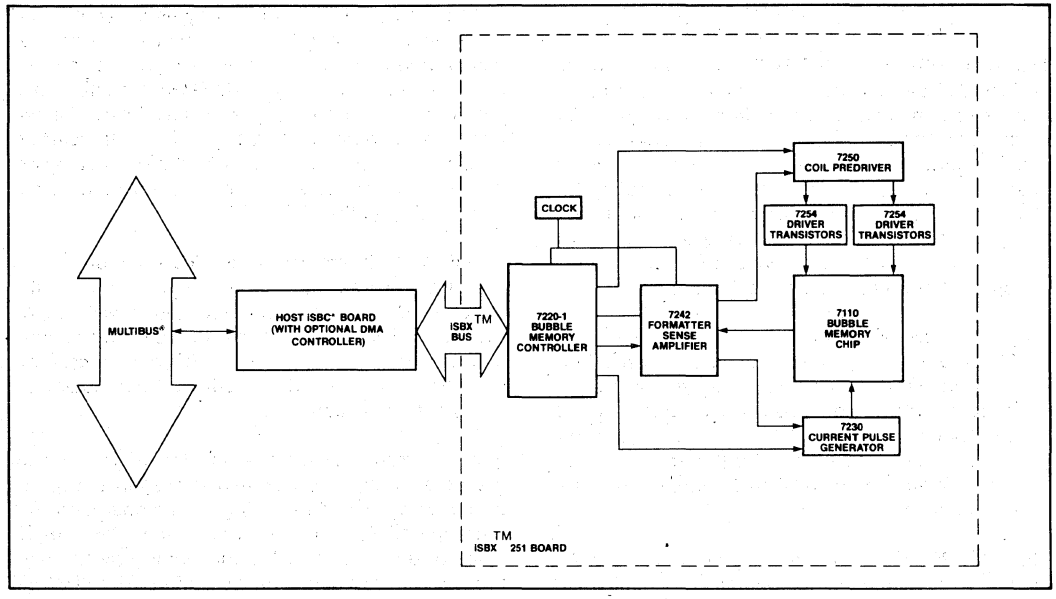


Figure 1. Block Diagram

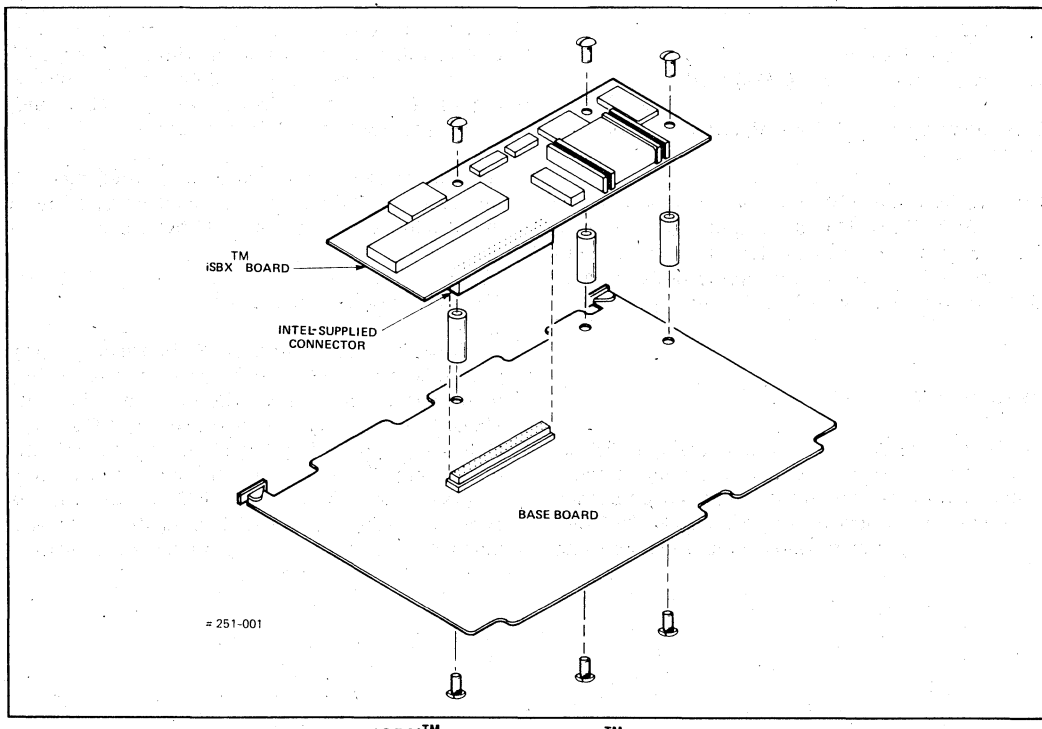


Figure 2. iSBX™ MULTIMODULE™ Board Concept

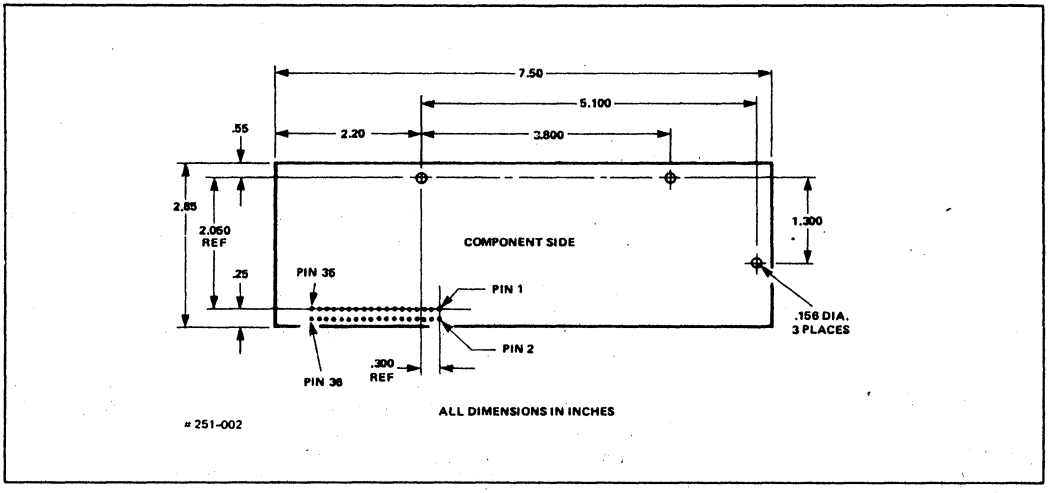


Figure 3. iSBX™ Connector and Spacer Locations

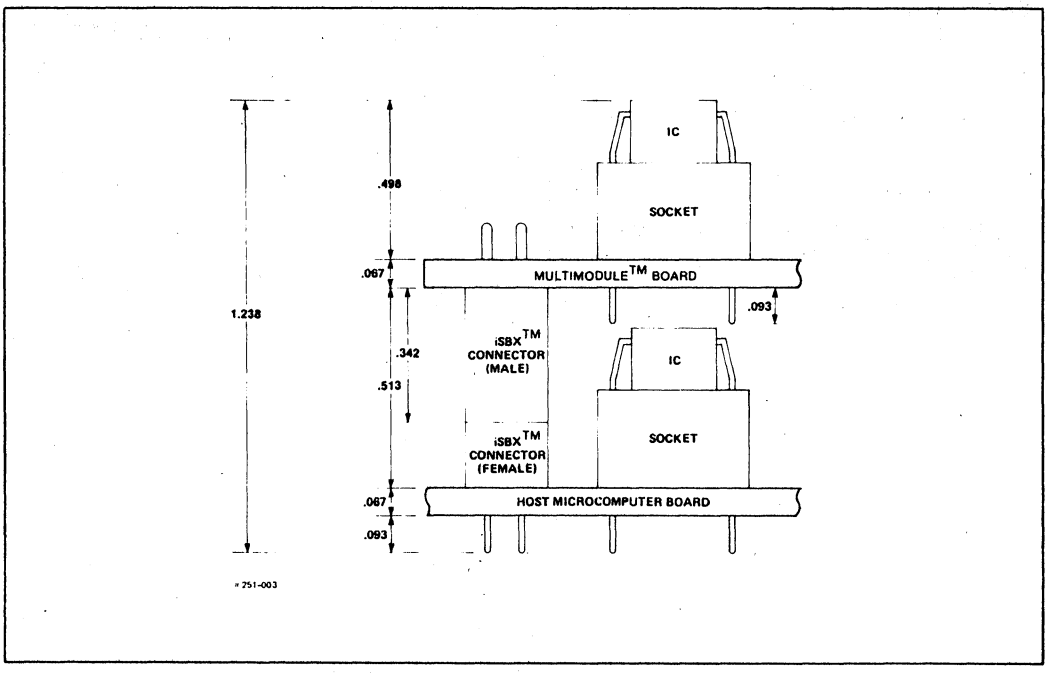


Figure 4. Mounting Clearances (Inches)

SPECIFICATIONS

Storage Capacity

—128K Eight-Bit Bytes
—2048 Pages
—Page Length:
64 bytes with ECC
68 bytes without ECC

Physical Characteristics

Width.....7.24 cm (2.85 in.)
Length.....19.05 cm (7.50 in.)
Height.....1.27 cm (0.498 in.)
Weight.....362.9 gm (12.8 oz.)

Environment

iSBX 251 board.....0-60°C Ambient
iSBX 251C board.....10°-40°C Ambient
Temperatures quoted are ambient with 100 lfm air flow.

Operational Modes

Polled, Interrupt Driven, or DMA (with Host DMA Controller)

Electrical Requirements

D.C. power, supplied through iSBX connector:

Voltage	D.C. Tolerance	Power Off/Power Fail Decay Rate	Maximum Current
+12 Volts	±5%	less than 1.10 volts/msec	400mA
+5 Volts	±5%	less than 0.45 volts/msec	365mA

Performance

Maximum Data Rate.....400K bits/sec
Average Access Time.....48 ms
Average Transfer Rate.....68K bits/sec

Interface Requirements

—TTL compatible
—iSBX 251 male connector plugs into 36-pin or 44-pin host female connector Intel No. 7906.

Relative Humidity:

0% to 95% without condensation

Non-Volatile Storage temperature:

iSBX 251C board.....– 20 to + 75°C
iSBX 251 board.....– 40 to + 90°C

Additional Documentation

iSBX251 Technical Manual (Order Number 112924)
iSBX Bus Specification (Order Number 142686)
Memory Components Handbook
(Order Number 210830)

*MDS is an ordering code and is not used as a product name or trademark. MDS is a registered trademark of Mohawk Data Sciences Corporation.

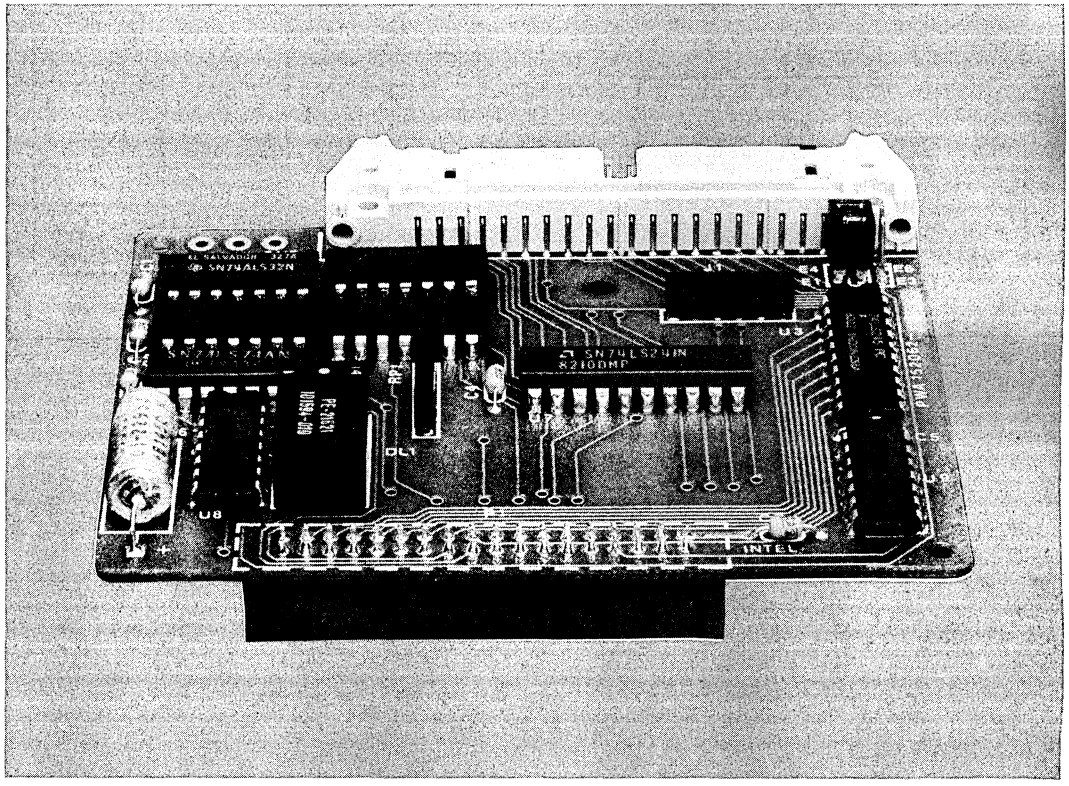
ADVANCE INFORMATION

iSBX™ 258
iSBX INTERFACE MULTIMODULE™ FOR INTEL BUBBLE CASSETTE SYSTEM

- **Interfaces iBC Bus to iSBX™ Bus**
- **Single-Wide iSBX™ MULTIMODULE™**
- **Drives One or Two iBC Systems**
- **Speeds Design Time with iBC Systems**

The iSBX 258 Interface MULTIMODULE for the Intel Bubble Cassette (iBC) System provides an iSBX interface for the iBC system. Each iSBX 258 can interface up to two daisychained iBC systems in polled or interrupt data transfer modes, or one in the DMA mode.

The iSBX 258 plugs into Intel iSBC® Single Board Computer products which have iSBX connectors or any other processor boards with iSBX connectors. It is included in the iBCK 12 iBC prototyping kit to facilitate design work on the iBC system.



JUNE 1984
ORDER NUMBER: 231085-001

FUNCTIONAL DESCRIPTION

The iSBX 258 is a completely assembled and tested iBC bus to iSBX bus interface board. (See Figure 1 for a block diagram of its recommended configuration in an iBC system.) It consists of buffer circuitry mounted on a single-wide iSBX MULTIMODULE card. It is completely iSBX bus compatible and allows easy iSBX interface to an iBC system with up to 18″ of ribbon cable.

OPERATION

The operation of the MULTIMODULE is software transparent to the user while maintaining all iSBX bus specifications; +5Vdc ±5% is supplied from the iSBX bus.

MOUNTING TECHNIQUE

As shown in Figure 2, the iSBX 258 plugs into a host board via the iSBX connector and is secured by a spacer with a screw. A single-wide iSBX MULTIMODULE is used and one Multibus card slot is occupied in addition to the card slot for the host board. Dimensions of the board and host board/MULTIMODULE height tolerances are given in Figures 3 and 4. Although the iSBX 258 board's male iSBX connector has the standard 36 pins, it will also plug into the expanded 44 pin female iSBX connector.

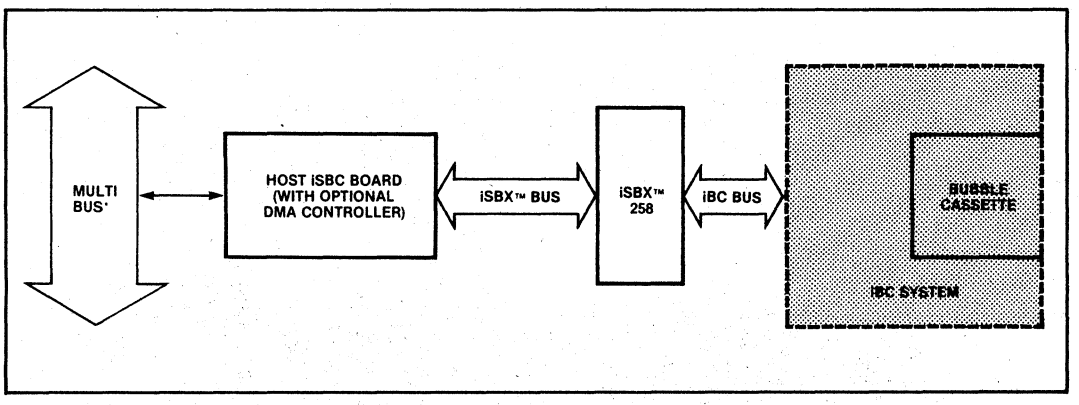


Figure 1. Block Diagram

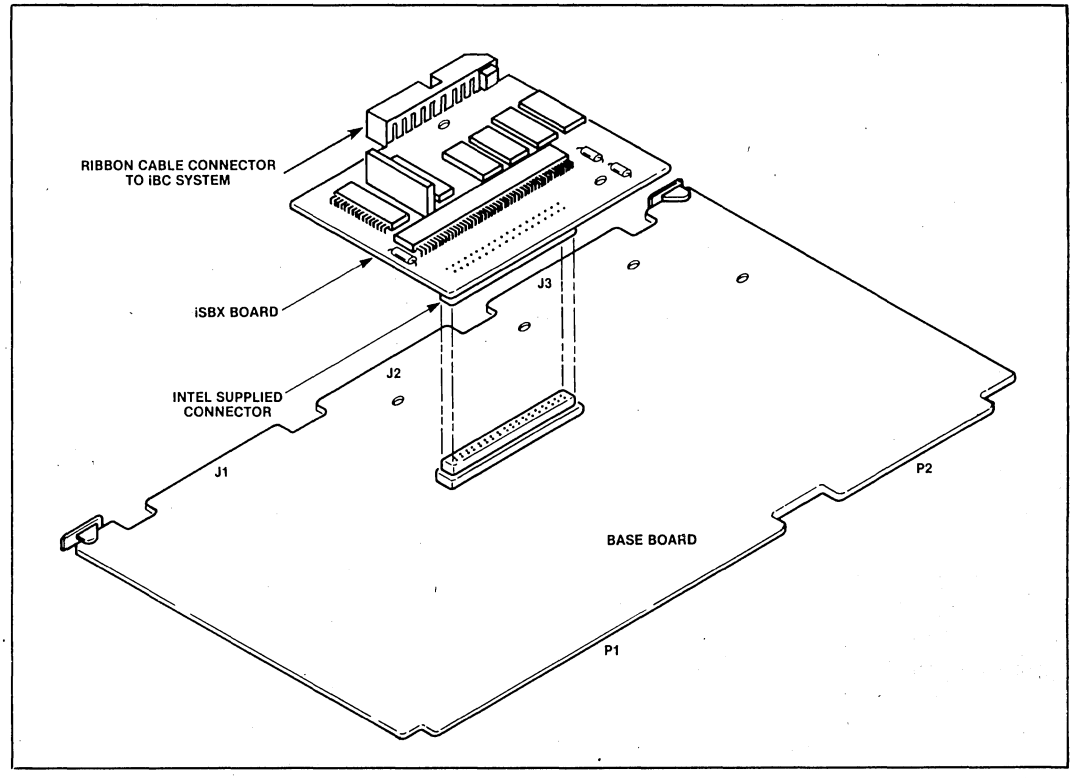


Figure 2. MULTIMODULE™ Mounting

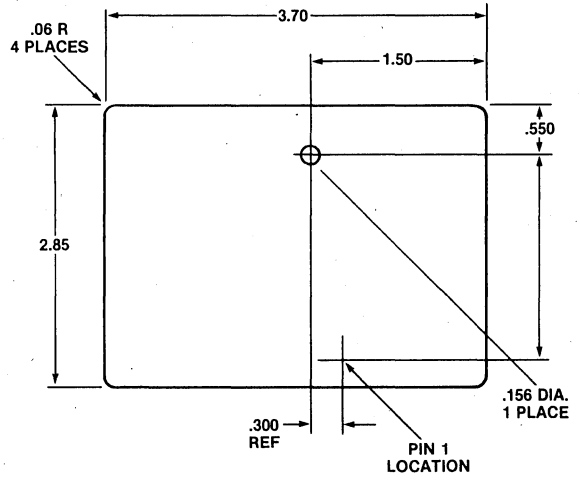


Figure 3. Dimensions

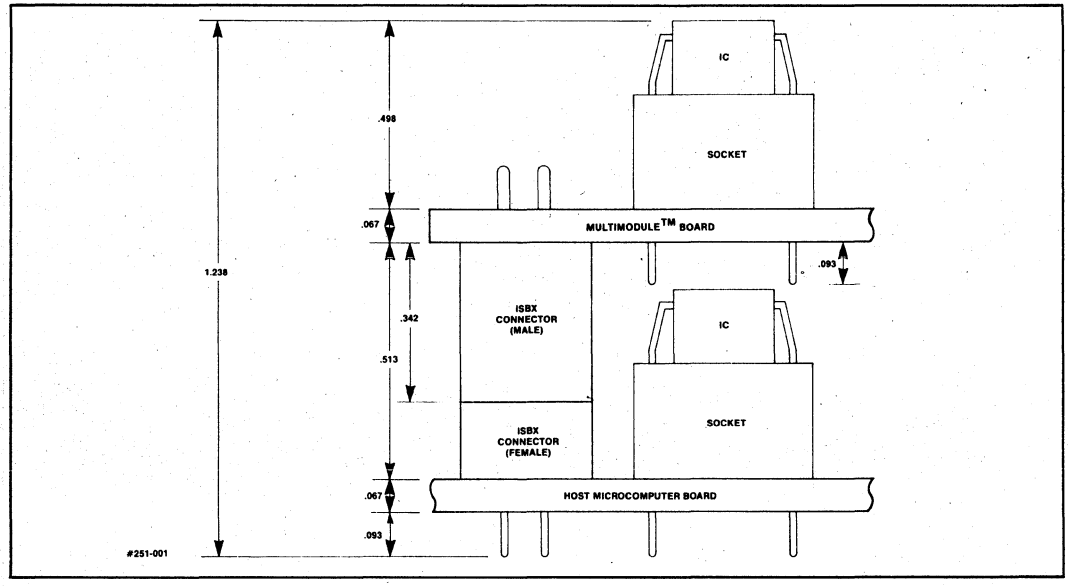


Figure 4. Mounting Clearance

SPECIFICATIONS

Physical Characteristics

Width: 7.24 cm (2.85 in.)
Length: 9.40 cm (3.70 in.)
Height: 2.05 cm (0.81 in.)
Weight: 1.8g

Environment

iSBX 258 Board 0-65°C
Temperatures are ambient in free moving air.

Operational Modes

Supports Polled, Interrupt-Driven, or DMA (with Host DMA controller) transfers with the iBC system.

Electrical Requirements

D.C. power, supplied through iSBX connector: +5Vdc ±5%, 285 mA (max).

Note: Three auxilliary points are provided which supply +5V, +12V and GND. Power available is:

V_{CC} = 5V @ 2.7A (max)
V_{dd} = 12V @ 1.0A (max)

Performance in iBC System

Maximum Data Rate: 100KBits/sec
Average Access Time: 48 msec.
Average Transfer Rate: 68KBits/sec

Interface Requirements

— TTL compatible
— iSBX 258 male iSBX connector plugs into 36-pin or 44-pin host female connector
— iSBX 258 40-pin male ribbon cable connector plugs into 40-pin female ribbon cable connector cabled to iBC system.

Relative Humidity

0% to 95% without condensation

Additional Documentation

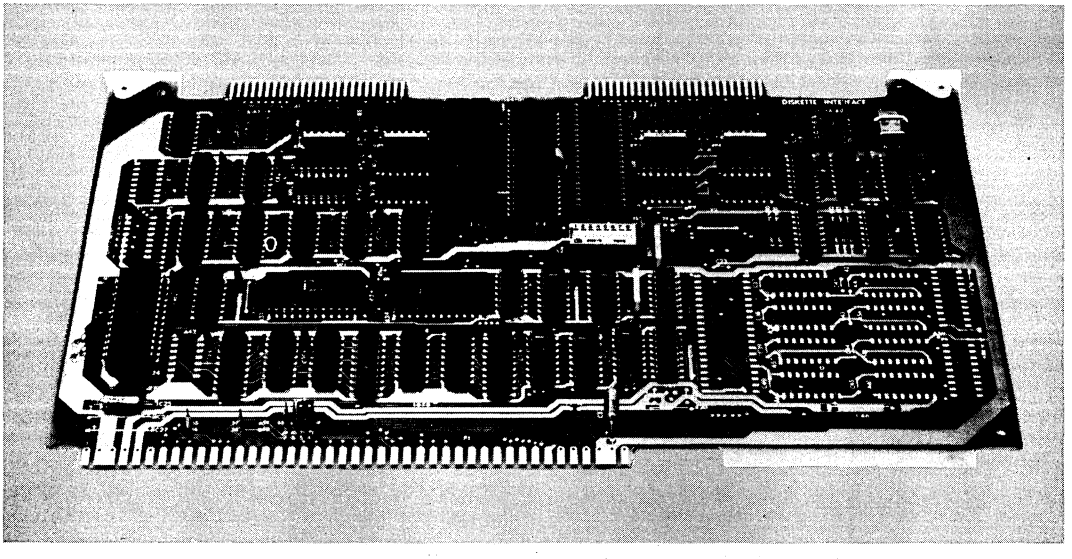
Intel Bubble Cassette System Users' Manual (Order #122278-001)
iSBX Bus Specification (Order #142686)

Peripheral Controllers 8

iSBC® 204
SINGLE DENSITY FLEXIBLE DISKETTE CONTROLLER

- **Full compatibility with iSBC® 80, iSBC® 86, and iSBC 88 Single Board Computers**
- **Direct compatibility with most single-density, soft-sectored standard- (8″) and mini-size (5¼″) flexible diskette drives**
- **Software supported by iRMX™ 80, iRMX™ 86 and iRMX™ 88 Real-Time Multi-tasking Executive disk file system**
- **Support by CP/M operating system**
- **DMA input/output allows single board computers to process in parallel with diskette transfer operations**
- **Programmable track-to-track access, head-settling, and head-load times**
- **On-board data separation logic**
- **Read, write, verify, and search on single or multiple sectors**
- **Single +5V supply**

The Intel iSBC 204 Single Density Flexible Diskette Controller is a single board universal diskette controller capable of supporting virtually any software-sectored, single density diskette drive. The standard iSBC 204 Controller can control two drive surfaces (two single-sided drives or one double-sided drive). With the addition of a second (optional) Intel 8271 component, up to four drives can be supported. In addition to the standard IBM 3740 formats, the controller supports sector lengths of up to 4096 bytes plus mini-size drive formats. The iSBC 204's wide range of drive compatibility is achieved without compromising performance. The operating characteristics (track-to-track access, head-load, and head-settling times) are specified under user program control. The controller can read, write, verify, and search either single or multiple sectors.



FUNCTIONAL DESCRIPTION

Intel's 8271 Floppy Disk Controller (FDC) circuit is the heart of the iSBC 204 Controller. On-board data separation logic performs standard FM encoding and decoding, obviating external separation circuitry at the drive. Diskette data transfers are DMA (direct memory access) through an on-board Intel 8257 DMA controller circuit which manages DMA transfers and signals the master iSBC processor on completion of the transfer. A block diagram of the iSBC 204 Controller is shown in Figure 1.

Universal Drive and MULTIBUS® Compatibility

Because the iSBC 204 Controller has universal drive compatibility, it can be used to control virtually any standard- or mini-sized single density diskette drive. Moreover, the iSBC 204 Controller fully supports the microcomputer industry standard MULTIBUS system bus and can be used with any single board computer or system compatible with Intel's bus. Because the iSBC 204 Controller is programmable, its performance is not compromised by its universal drive compatibility. The track-to-track access, head-load, and head-settling characteristics of the selected drive model are program specified. Data may be organized in a fully compatible IBM 3740 sector format, in sectors up to 4096 bytes in length, or in formats compatible with the mini-sized diskette drives.

Interface Characteristics

Expandability — Each standard iSBC 204 Controller includes a single 8271 FDC circuit capable of supporting two drive surfaces. Optionally the iSBC 204 may be expanded to support four single-sided (or two double-sided) drives with the insertion of a second 8271 component into an on-board socket.

Simplified Interface —The cables between the iSBC 204 Controller and the drive(s) may be either low cost, flat ribbon cable with mass termination connectors or twisted pair conductors with individually wired connectors. An on-board, cross-connect matrix allows optional drive control and status signals to be connected while maintaining pin-to-pin compatibility.

Programming

The powerful 8271 FDC circuit is capable of executing high-level commands that simplify system software development. The device can read, write, and verify both single and multiple sectors. CRC characters are generated and checked automatically. Up to two tracks on each surface may be designated "bad" and logically removed from the diskette.

Sector Scanning — Scan commands permit sectors to be searched for a specified data pattern or "key". During scan operations the pattern image from memory is continuously compared with a sector or multiple sectors

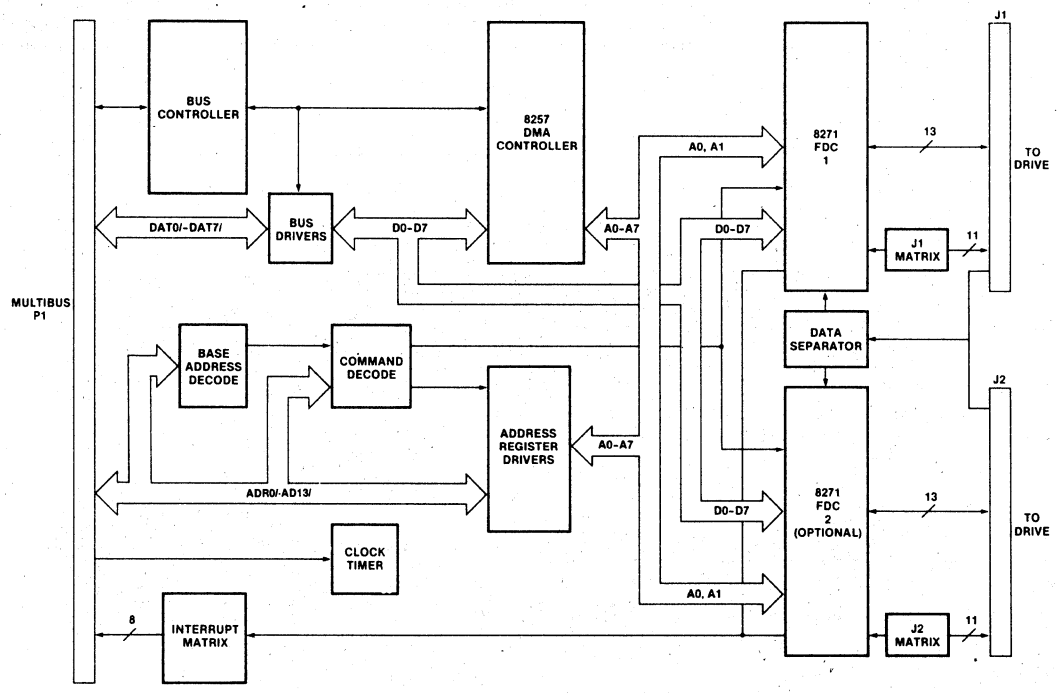


Figure 1. iSBC® 204 Single Density Diskette Controller Block Diagram

AFN-01279A

read from the diskette. No CPU intervention is required until a match is found or all specified sectors have been searched.

Program Initiation — All diskette operations are initiated by standard input/output (I/O) port operations through an iSBC single board computer. System software first initializes the controller with the operating characteristics of the selected drive. The diskette is then formatted under program control. For subsequent transfers, the starting memory address and transfer mode are specified for the DMA controller. Data transfers occur in response to commands output by the CPU.

Data Transfer — Once a diskette transfer operation has been initiated, the controller acts as a bus master and transfers data over the MULTIBUS at high speed. No CPU intervention is required until the transfer is complete as indicated either by the generation of an interrupt on the bus or by examination of a "done" bit by the CPU.

SPECIFICATIONS

Compatibility

CPU — Any iSBC MULTIBUS computer or system mainframe.

Drive — Single density, standard- (8") and mini-sized (5¼") diskette drives. The standard iSBC 204 Controller supports two single-sided drives or one double-sided drive. By adding an (optional) 8271 FDC, four single-sided or two double-sided drives may be supported. The following drives are known to be compatible:

Standard Size	Mini Size
CDC 9404	PERTEC FD200
GSI 110	SHUGART SA400
MEMOREX 550	WANGCO 82
MEMOREX 552 (dual-sided)	
SHUGART 800	
SHUGART 850 (dual-sided)	
WANGCO 76S	
PERTEC 650 (SD/DD, DBL. Head)	

Diskette — Unformatted IBM Diskette 1 (or equivalent single-sided); unformatted IBM Diskette 2 (or equivalent double-sided); unformatted Shugart SA104 Diskette (or equivalent mini).

Data Organization and Capacity

(Standard Size Drives)

	IBM Format			Non-IBM Format		
Bytes per sector	128	256	512	1024	2048	4096
Sectors per track	26	15	8	4	2	1
Tracks per diskette	77			Up to 255		
Bytes per diskette (77 tracks)	256,256 (128-byte sector) 295,680 (256-byte sector) 315,392 (512-byte sector)			315,392		

Drive Characteristics

	Standard Size	Mini Size
Transfer rate (KB/sec)	250	125
Disk speed (RPM)	360	300
Track-to-track access (programmable)	1 to 255 ms in 1 ms steps	2 to 510 ms in 2 ms steps
Head settling time (programmable)	0 to 255 ms in 1 ms steps	0 to 510 ma in 2 ms steps
Head load time (programmable)	0 to 60 ms in 4 ms steps	0 to 120 ms in 8 ms steps

Equipment Supplied

iSBC 204 Controller

Reference Schematic

Controller-to-drive cabling and connectors are not supplied with the iSBC 204 Controller. Cables can be fabricated easily using either flat ribbon cable or twisted pair conductors with commercially available connectors as described in the iSBC 204 Hardware Reference Manual.

Optional Equipment

8271 Flexible Diskette Controller Component — Adding a second 8271 device to the fully tested circuit on the iSBC 204 Controller allows four drive surfaces to be supported.

Physical Characteristics

Width — 6.75 in. (17.15 cm)
Height — 0.5 in. (1.27 cm)
Length — 12.0 in. (30.48 cm)
Shipping Weight — 1.75 lb (0.80 kg)
Mounting — Occupies one slot of iSBC system chassis or iSBC 604/614 cardcage.

Electrical Characteristics

Power Requirements — 5.0V (±5%), 2.5A max

Environmental Characteristics

Temperature — 0°C to 55°C (operating); −55°C to +85°C (non-operating)

Humidity — Up to 90% relative humidity without condensation (operating); all conditions without condensation or frost (non-operating)

Reference Manuals

9800568 — iSBC 204 Diskette Controller Hardware Reference Manual (NOT SUPPLIED).

9800522 — RMX/80 User's Guide (NOT SUPPLIED).

Reference manuals are shipped with each product only if designated SUPPLIED (see above). Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

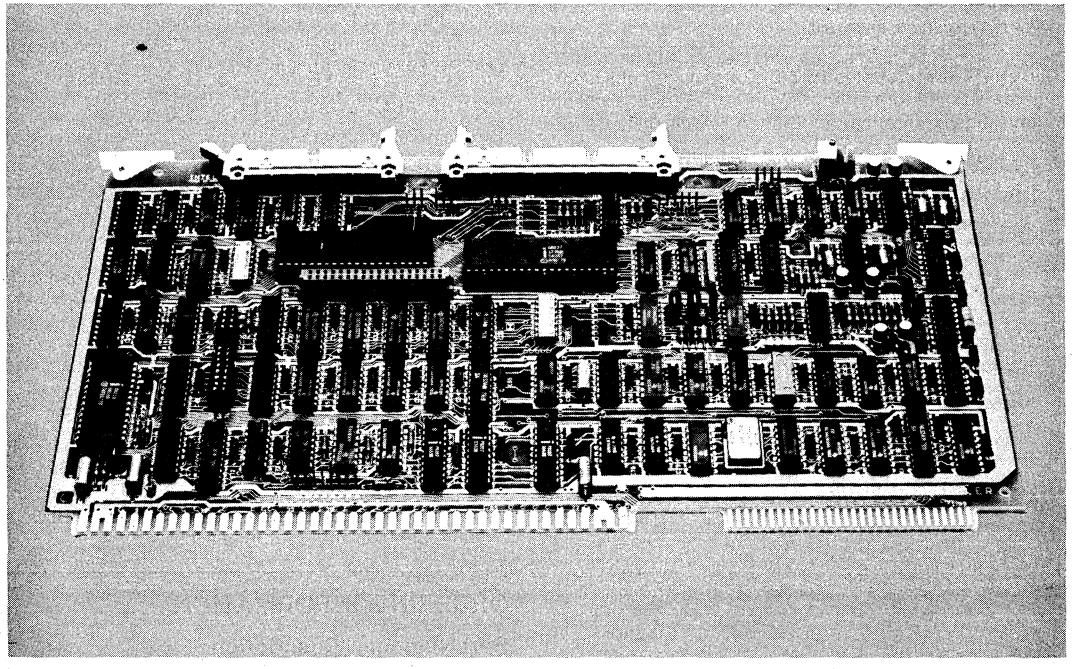
ORDERING INFORMATION

Part Number	Description
SBC 204	Universal Flexible Diskette Controller

iSBC® 208
FLEXIBLE DISKETTE CONTROLLER

- **Compatible with all iSBC ®80, iSBC® 86, and iSBC® 88 Single Board Computers**
- **Controls most single and double density diskette drives**
- **On-board iSBX™ bus for additional functions**
- **User-programmable drive parameters allow wide choice of drives**
- **Phase lock loop data separator assures maximum data integrity**
- **Read and write on single or multiple sectors**
- **Single +5V Supply**
- **Capable of addressing 16M bytes of system memory**

The Intel iSBC 208 Flexible Disk Controller is a diskette controller capable of supporting virtually any soft-sectored, double density or single density diskette drive. The standard controller can control up to four drives with up to eight surfaces. In addition to the standard IBM 3740 formats and IBM System 34 formats, the controller supports sector lengths of up to 8192 bytes. The iSBC 208 board's wide range of drive compatibility is achieved without compromising performance. The operating characteristics are specified under user program control. The controller can read, write, verify, and search either single or multiple sectors. Additional capability such as parallel or serial I/O or special math functions can be placed on the iSBC 208 board by utilizing the iSBX bus connection.



FUNCTIONAL DESCRIPTION

Intel's 8272 Floppy Disk Controller (FDC) circuit is the heart of the iSBC 208 Controller. On-board data separation logic performs standard MFM (double density) and FM (single density) encoding and decoding, eliminating the need for external separation circuitry at the drive. Data transfers between the controller and memory are managed by a DMA device which completely controls transfers over the MULTIBUS system bus. A block diagram of the iSBC 208 Controller is shown in Figure 1.

Universal Drives and the iSBC® 208 Controller

Because the iSBC 208 Controller has universal drive compatibility, it can be used to control virtually any standard- or mini-sized diskette drive. Moreover, the iSBC 208 Controller fully supports the iSBX bus and can be used with any iSBX module compatible with this bus. Because the iSBC 208 Controller is programmable, its performance is not compromised by its universal drive compatibility. The track-to-track access, head-load, and head-unload characteristics of the selected drive model are program specified. Data may be organized in sectors up to 8192 bytes in length.

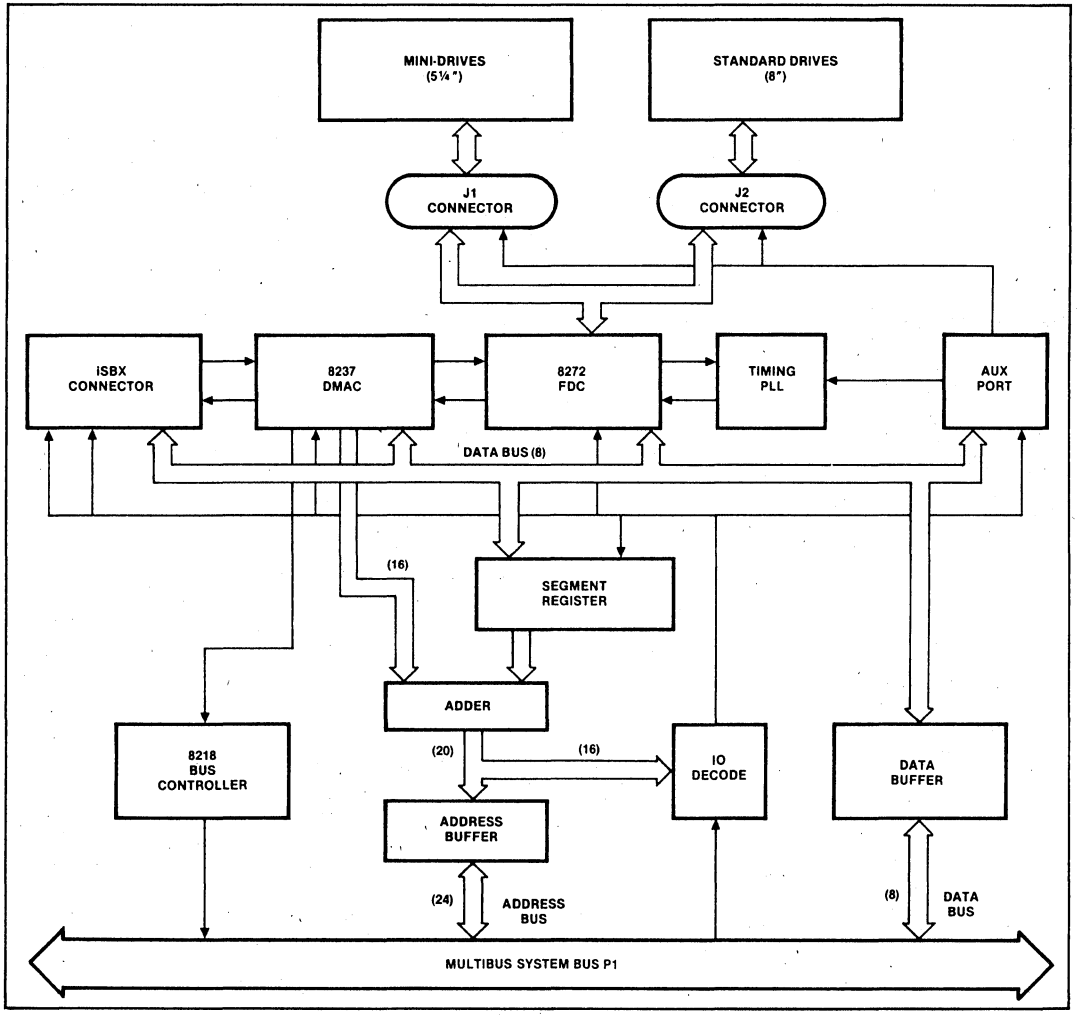


Figure 1. iSBC ®208 Flexible Disk Controller Block Diagram

AFN-01629A

Interface Characteristics

The standard iSBC 208 Controller includes an Intel 8272 Floppy Disk Controller chip which supports up to four drives, single or double sided.

SIMPLIFIED INTERFACE—The cables between the iSBC 208 Controller and the drive(s) may be low cost, flat ribbon cable with mass termination connectors. The mechanical interface to the board is a right-angle header with locking tabs for security of connection.

PROGRAMMING — The powerful 8272 FDC circuit is capable of executing high-level commands that simplify system software development. The device can read and write both single and multiple sectors. CRC characters are generated and checked automatically. Recording density is selected at each Read and Write to support the industry standard technique of recording basic media information on Track 0 of Side 0 in single density, and then switching to double density (if necessary) for operations on other tracks.

Program Initiation—All diskette operations are initiated by standard input/output (I/O) port operations through an iSBC single board computer. System software first initializes the controller with the operating characteristics of the selected drive. The diskette is then formatted under program control. For subsequent transfers, the starting memory address and transfer mode are specified for the DMA controller. Data transfers occur in response to commands output by the CPU.

Data Transfer—Once a diskette transfer operation has been initiated, the controller acts as a bus master and transfers data over the MULTIBUS at high speed. No CPU intervention is required until the transfer is complete as indicated either by the generation of an interrupt on the bus or by examination of a "done" bit by the CPU.

iSBX BUS SUPPORT — One connector is available on the iSBC 208 board which supports the iSBX system bus. This connector supports single-byte transfer as well as higher-speed transfers supervised by the DMA controller. Transfers may take place in polled or interrupt modes, user-selected. The presence of the iSBX bus allows many different functions to be added to the board. Serial I/O, parallel I/O and various special-purpose math functions are only a few of the capabilities available on iSBX MULTIMODULE boards.

SPECIFICATIONS

Compatibility

CPU—Any iSBC MULTIBUS computer or system main frame

Devices—Double or single density standard (8") and mini (5¼") flexible disk drives. The drives may be single or double sided. Drives known to be compatible are:

Standard (8")		Mini (5¼")	
Caldisk	143M	Shugart	450 SA 400
Remex	RFD 4000	Micropolis	1015-IV
Memorex	550	Pertec	250
MFE	700	Siemens	200-5
Siemens	FDD 200-8	Tandon	TM-100
Shugart	SA 850/800	CDC	9409
Pertec	FD 650	MPI	51/52/91/92
CDC	9406-3		

Diskette—Unformatted IBM Diskette 1 (or equivalent single-sided media); unformatted IBM Diskette 2D (or equivalent double-sided).

Equipment Supplied

iSBC 208 Controller
Reference Schematic
Controller-to-drive cabling and connectors are not supplied with the controller. Cables can be fabricated with flat cable and commercially-available connectors as described in the iSBC 208 Hardware Reference Manual

Physical Characteristics

Width—6.75 inches (17.15 cm)
Height—0.5 inches (1.27 cm)
Length—12.0 inches (30.48 cm)
Shipping Weight—1.75 pounds (0.80 Kg)
Mounting—Occupies one slot of iSBC system chassis or iSBC 604/614 Cardcage/Backplane. With an iSBX MULTIMODULE board mounted, vertical height increases to 1.13 inches (2.87 cm).

Electrical Characteristics

Power Requirements— +5 VDC @ 3.0A

Data Organization and Capacity

Standard Size Drives

	Double Density						Single Density					
	IBM System 34			Non-IBM			IBM System 3740			Non-IBM		
Bytes per Sector	256	512	1024	2048	4096	8192	128	256	512	1024	2048	4096
Sectors per Track	26	15	8	4	2	1	26	15	8	4	2	1
Tracks per Diskette	77			256			77			256		
Bytes per Diskette (Formatted, per diskette surface)	512,512 (256 bytes/sector) 591,360 (512 bytes/sector) 630,784 (1024 bytes/sector)			630,784			256,256 (128 byte/sector) 295,680 (256 bytes/sector) 315,392 (512 bytes/sector)			315,392		

Drive Characteristics

	Standard Size	Mini Size
	Double/Single Density	Double/Single Density
Transfer Rate (K bytes/sec)	62.5/31.25	31.25/15.63
Disk Speed (RPM)	360	300
Step Rate Time (Programmable)	1 to 16 msec/track in 1 msec increments	2 to 32 msec/track in 2 msec increments
Head Load Time (Programmable)	2 to 254 msec in 2 msec increments	4 to 508 msec in 4 msec increments
Head Unload Time (Programmable)	16 to 240 msec in 16 msec increments	32 to 480 msec in 32 msec increments

Environmental Characteristics

Temperature—0°C to 55°C (operating); −55°C to +85°C (non-operating)

Humidity—Up to 90% Relative Humidity without condensation (operating); all conditions without condensation or frost (non-operating)

Reference Manual

143078-001—iSBC 208 Flexible Disk Controller Hardware Reference Manual (NOT SUPPLIED). Reference manuals may be ordered from any Intel sales representative, distributor office, or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, CA 95051.

ORDERING INFORMATION

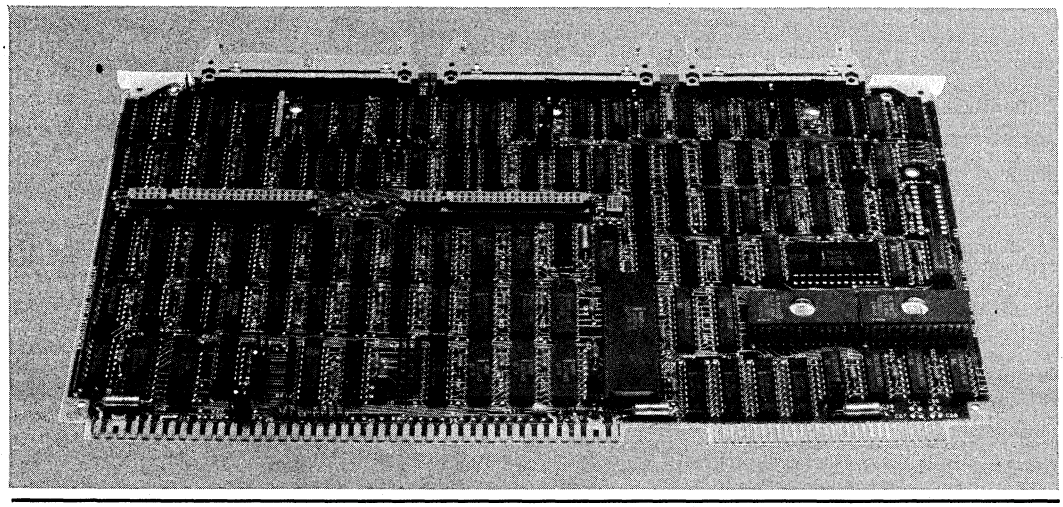
Part Number	Description
SBC 208	Flexible Disk Controller

iSBC® 215 GENERIC WINCHESTER CONTROLLER

- **Controls up to four 5¼", 8" or 14" Winchester disk drives from over ten different vendors**
- **Compatible with Industry Standard MULTIBUS® (IEEE 796) Interface**
- **Suports ANSI X3T9/1226 standard interface**
- **Software drivers available for iRMX™ 86, iRMX™ 88 and Xenix* Operating Systems**
- **Intel 8089 I/O Processor provides intelligent DMA capability**
- **On-board diagnostics and ECC**
- **Full sector buffering on-board**
- **Capable of directly addressing 16 MB of system memory**
- **Removable back-up storage available through the iSBX™ 218A Flexible Disk Controller and the iSBX™ 217C ¼" Tape Interface Module†**

Using VLSI technology, the iSBC 215 Generic Winchester Controller (GWC) combines three popular Winchester controllers onto one MULTIBUS board: the iSBC 215A open loop controller, the iSBC 215B closed loop controller, and an ANSI X3T9/1226 standard interface controller. The combined functionality of the iSBC 215 Generic Controller supports up to four 5¼", 8" or 14" Winchester drives from over 10 different drive vendors. Integrated back-up is available via two iSBX MULTIMODULE boards; the iSBX 218A module for floppy disk drives and the iSBX 217C module for ¼" tape units.†

From the MULTIBUS side, the iSBC 215 GWC appears as one standard software interface, regardless of the drive type used. In short, the iSBC 215 GWC allows its user to change drive types without rewriting software. The iSBC 215 Generic Controller is totally downward compatible with its predecessors, the iSBC 215A and 215B controller; allowing existing iSBC 215A and 215B users to move quickly to the more powerful iSBC 215 Generic Winchester Controller. In addition, the iSBC 215 GWC directly addresses up to 16 megabytes of system memory.



OCTOBER 1984
ORDER NUMBER: 210618-002

FUNCTIONAL DESCRIPTION

Disk Interface

The iSBC 215 Generic Winchester Controller can interface to over 10 different disk drives. To change drive types the user need only reconfigure a minimal number of board jumpers and, if required, insert the proper formatting information into the command parameter blocks.

The ANSI X3T9/1226 standard interface is a simple one-for-one flat cable connection from drive to controller.

Full On-Board Buffer

The iSBC 215 Generic controller contains enough on-board RAM for buffering one full data sector. The controller is designed to make use of this buffer in all transfers. The on-board sector buffer prevents data overrun errors and allows the iSBC 215 Generic Winchester Controller to occupy any priority slot on the MULTIBUS.

ECC

High data integrity is provided by on-board Error Checking Code (ECC) logic. When writing sector ID or data fields, a 32-bit ECC, for burst error correction, is appended to the field by the controller. During a read operation, the same logic regenerates the ECC polynomial to the appended ECC. The ECC logic can detect an erroneous data burst up to 32 bits in length and using an 8089 algorithm can correct an erroneous burst up to 11 bits in length.

iSBX™ Interface

Two iSBX bus connectors provide I/O expansion capability for the iSBC 215 GWC. With the optional addition of the iSBX 218A Flexible Disk Controller MULTIMODULE™ and or the iSBX 217C ¼" Tape Interface Module, the iSBC 215 GWC can be configured into one of four types of peripheral subsystems, see Table 1.

Table 1. Peripheral Subsystem Configurations

	iSBC® 215	iSBX™ 218A	iSBX™ 217C[1]
Winchester Only	✓		
Winchester + Floppy	✓	✓	
Winchester + ¼ "Tape	✓		✓
Winchester + Floppy + ¼ "Tape	✓	✓	✓

Expanded I/O Capability

The iSBC 215 GWC controller allows the execution of user-written 8089 programs located in on-board or MULTIBUS system RAM. Thus the full capability of the 8089 I/O processor can be utilized for custom I/O requirements.

MULTIBUS® Interface

The iSBC 215 Generic Controller interfaces to the system CPU(s) through MULTIBUS memory. The iSBC 215 Generic controller directly addresses 16

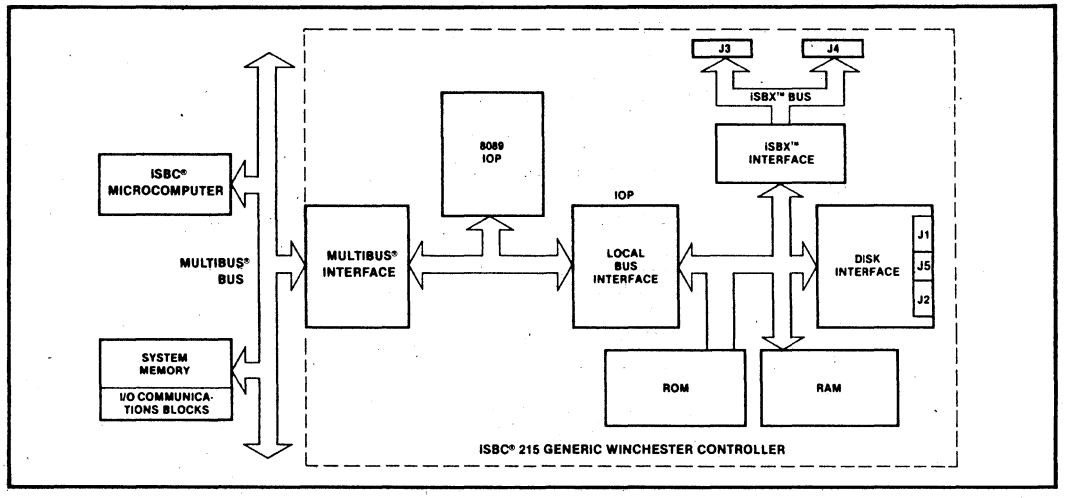
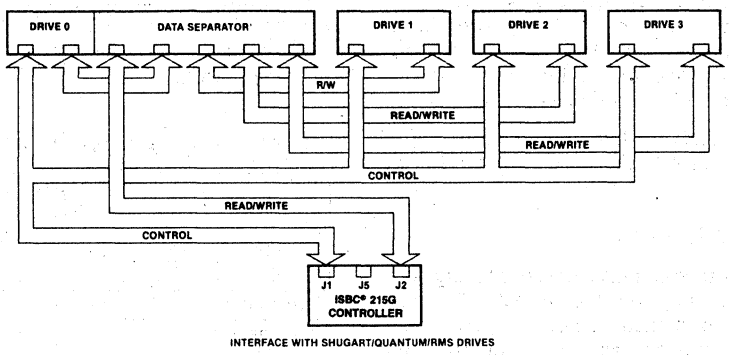


Figure 1. Block Diagram of iSBC® 215 Generic Winchester Disk Controller

megabytes of system memory. Commands are passed to and from the iSBC 215 GWC via memory based parameter blocks; these parameter blocks are executed.

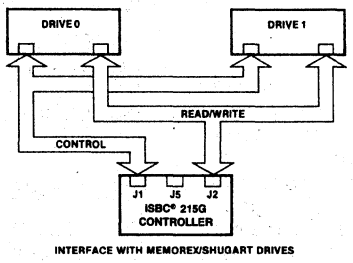
directly by the iSBC 215 GWC thus off-loading the system CPU(s). Data transfers to and from the iSBC 215 GWC are done via the high speed DMA capability of the Intel 8089 I/O processor.



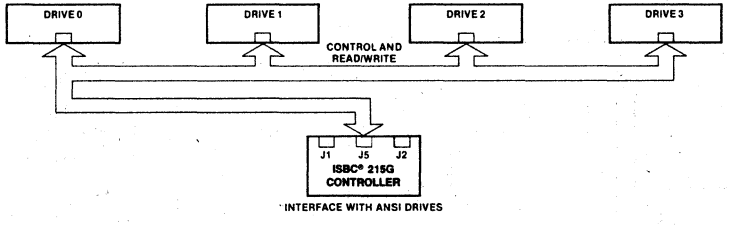
INTERFACE WITH SHUGART/QUANTUM/RMS DRIVES

NOTE:

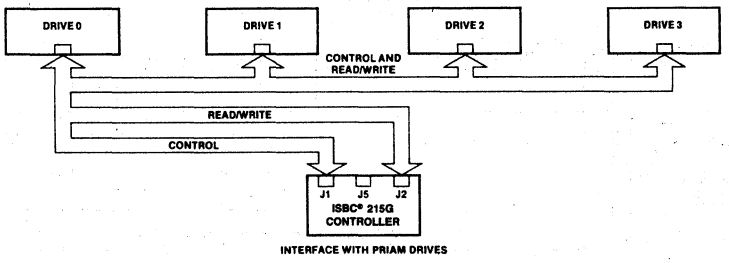
1. Shugart SA1000 or RMS Data Express.* *Data Express is a trademark of Rotating Memory Systems.



INTERFACE WITH MEMOREX/SHUGART DRIVES



INTERFACE WITH ANSI DRIVES



INTERFACE WITH PRIAM DRIVES

Figure 2. Controller to Drive Interfacing

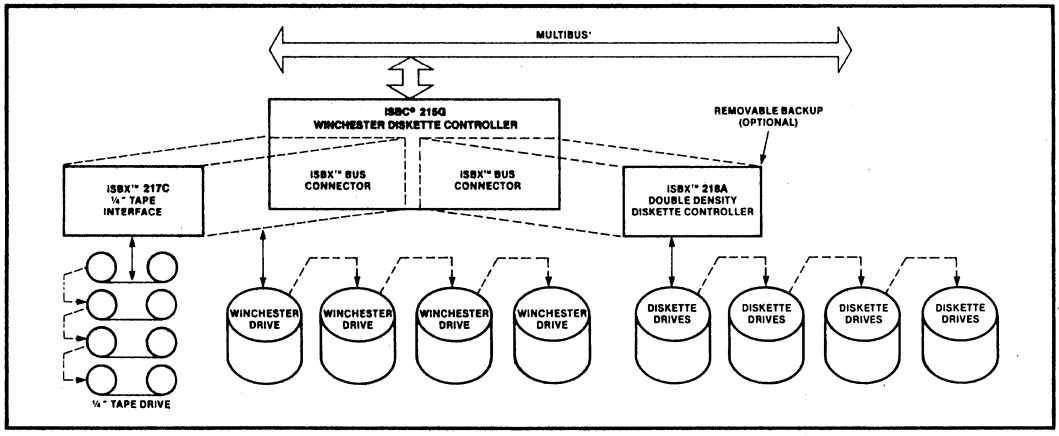


Figure 3. Subsystem Configuration (with Optional Diskette Backup)

SPECIFICATIONS

Compatibility

CPU — Any iSBC MULTIBUS computer or system mainframe.

Disk Drives — Winchester Disk Drives; both open-loop and closed-loop head positioner types. The following drives are known to be compatible:

Open-Loop
Shugart SA 1000 Series Shugart SA 4000 Series Memorex 100 Series Quantum Q2000 Series Fujitsu 2301, 2302 CDC 9410 RMS 5¼" Series Rodine 5¼" Series Ampex 5¼" Series CMI 5¼" Series
Closed-Loop
Priam 8" and 14" Drive Series
ANSI
3M 8430 Series Kennedy 6170 Series Micropolis 8" Series Pertec Trackstar Series Priam 8" Series Megavault (SLI) 8" Series
iSBX™ MULTIMODULE™ Boards
iSBX™ 218A Flexible Disk Controller iSBX™ 217C ¼" Tape Interface

Equipment Supplied

iSBC 215 Generic Winchester Controller Reference Schematic

Controller-to-drive cabling and connectors are not supplied with the controller. Cables can be fabricated with flat cable and commercially-available connectors as described in the iSBC 215G Hardware Reference Manual.

Physical Characteristics

Width — 6.75 in. (17.15 cm)

Height — 0.5 in. (1.27 cm)

Length — 12.0 in. (30.48 cm)

Shipping Weight — 19 oz. (.54 kg)

Mounting — Occupies one slot of iSBC system chassis or cardcage/backplane

With an iSBX MULTIMODULE board mounted, vertical height increases to 1.13 in. (2.87 cm).

Electrical Characteristics

Power Requirements

+5 VDC@4.52A max
−5 VDC@0.015A max[1]
+12 VDC@0.15A max[2]
−12 VDC@0.055A max[1,2]

Notes:

1. On-board regulator and jumper allows −12 VDC usage from MULTIBUS.
2. Required for some iSBX MULTIMODULE boards.

Data Organization

Bytes/Sector	Sectors/Track[1] 128	256	512	1024
Priam 8″	72	42	23	12
Priam 14″	107	63	35	18
RMS/Shugart 8″/Quantum/Ampes/Rodine/CM1	54	31	17	9
Fujitsu/Memorex	64	38	21	11
Shugart 14″	96	57	31	16
CDC Finch	64	41	23	12
3M (ANSI)	82	51	29	16
Megavault (ANSI)	73	43	21	12
Kennedy (ANSI)	74	43	23	12
Micropolis (ANSI)	71	44	25	13
Pertec (ANSI)	85	52	29	15

NOTES:
1. Maximum allowable for corresponding selection of bytes per sector.

Drives per Controller

5¼″ Winchester Disk Drives — Up to four RMS, CMI, Rodine or Ampex drives.

8″ Winchester Disk Drives — Up to four ANSI, Shugart, Quantum or Priam drives; up to two Memorex, CDC, or Fujitsu drives.

14″ Winchester Disk Drives — Up to four Priam drivers; up to two Shugart drives.

Flexible Disk Drives — Up to four drives through the optional iSBX 218A Flexible Disk Controller connected to the iSBC 215 GWC board's iSBX connector.

¼″ Tape Drives — Up to four drives through the optional iSBX 217C ¼″ Tape Interface Module connected to the iSBC 215 GWC board's iSBX connector.

Environmental Characteristics

Temperature — 0° to 55°C (operating); −55°C to +85°C (non-operating)

Humidity — Up to 90% relative humidity without condensation (operating); all conditions without condensation or frost (non-operating)

Reference Manual

144780 — iSBC 215 Generic Winchester Controller Hardware Reference Manual (NOT SUPPLIED)

Reference manuals may be ordered from any Intel sales representative, distributor office, or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, CA 95051

ORDERING INFORMATION

Part Number	Description
SBC 215G	Generic Winchester Controller

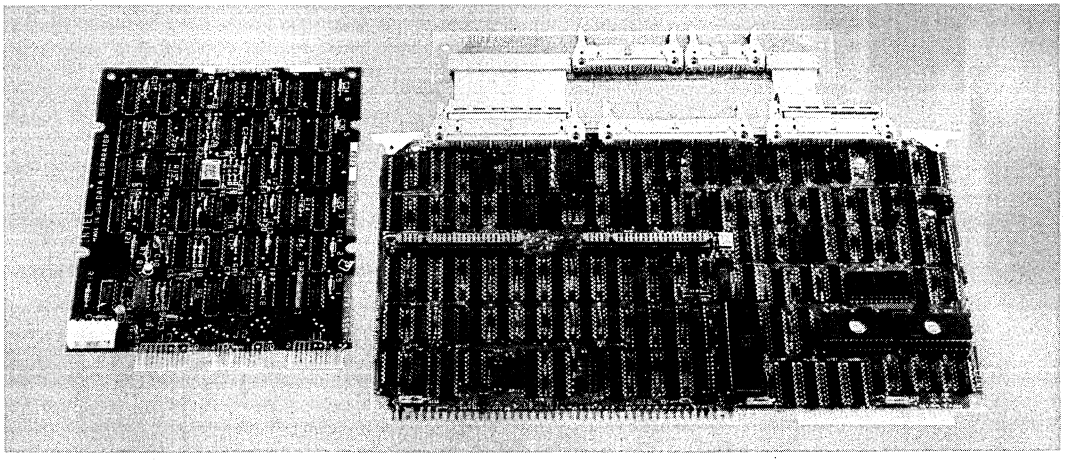
iSBC® 215G
5.25″ WINCHESTER CONTROLLER KIT

- **Includes iSBC® 215 Generic Winchester Controller and iSBC® 213 Data Separator Kit**
- **Controls up to two disk drives compatible with industry standard ST506/412 interface**
- **Data Separator board is form factor compatible to mount on 5.25″ drive**
- **Software drivers available for iRMX™ 86, iRMX™ 88 and XENIX* Operating Systems**
- **Intel 8089 I/O Processor provides intelligent DMA capability**
- **On-board diagnostics and ECC**
- **Full sector buffering on-board**
- **Capable of directly addressing 16 MB of system memory**
- **Removeable back-up storage available through the iSBX™ 218A Floppy Disk Controller and the iSBX™ 217C 1/4″ Tape Controller**

The Intel iSBC® 215 Generic Winchester Controller Kit supports up to two 5.25″ Winchester disk drives, compatible with the industry standard Seagate ST506/412 interface, which support buffered step pulses. The Kit consists of Intel's iSBC 215 Generic Winchester Controller (GWC) and the iSBC 213 Data Separator Kit. The data separator is designed to handle data transfers to 5 megabits per second and uses MFM (modified frequency modulation) data encoding/decoding for writing/reading on the drives. A scrambler card matches the pinout of the iSBC 215 Generic Winchester Controller to the pinout of ST506/412 compatible drives.

Optional integrated backup is available via two iSBX™ connectors on the iSBC 215 GWC and two iSBX modules: the iSBX 218A Floppy Disk Controller module and the iSBX 217C 1/4″ Tape Controller module.

The iSBC 215G 5.25″ Winchester Controller Kit is used in Intel's popular System 86/310 product.



* XENIX is a trademark of Microsoft Corporation.

OCTOBER 1984
ORDER NUMBER: 230969-002

FUNCTIONAL DESCRIPTION

Disk Interface

The iSBC 215G 5.25″ Winchester Controller Kit consists of the iSBC 215 Generic Winchester Controller and the iSBC 213 Data Separator Kit. The iSBC 213 Kit consists of a data separator board and a scrambler card and allows the iSBC 215 Generic Winchester Controller to interface to two 5.25″ Winchester drives utilizing the Seagate ST506/412 interface. The data separator converts data from MFM (modified frequency modulation) format to NRZ (non-return to zero) format for reading from Winchester disk drives and from NRZ format to MFM format for writing on Winchester disk drives. The data separator board also provides write precompensation, address mark generation/detection and a multiplexer that allows interfacing to two disk drives. The scrambler card matches the pinout of the iSBC 215 Generic Winchester Controller to the pinout required for the ST506/412 interface. A typical configuration using two 5.25″ Winchester drives and the iSBC 215G Winchester Controller Kit is shown in Figure 1.

The cables that connect the iSBC 215 Generic Winchester Controller to the scrambler card are included. The customer provides the cables going to the drives and between the scrambler board and the data separator board. These may be made with low cost, mass terminated, flat ribbon cable.

Full On-Board Buffer

The iSBC 215 GWC contains enough on-board RAM for buffering one full data sector. The controller is designed to make use of this buffer in all transfers. The on-board sector buffer prevents data overrun errors and allows the iSBC 215 Generic Winchester Controller to occupy any priority slot on the MULTIBUS® System Bus.

ECC

High data integrity is provided by on-board Error Checking Code (ECC) logic. When writing sector ID or data fields, a 32-bit ECC, for burst error correction, is appended to the field by the controller. During a read operation, the same logic regenerates the ECC polynominal and compares this second polynominal to the appended ECC. The ECC logic can detect an erroneous data burst up to 32 bits in length and using an 8089 algorithm

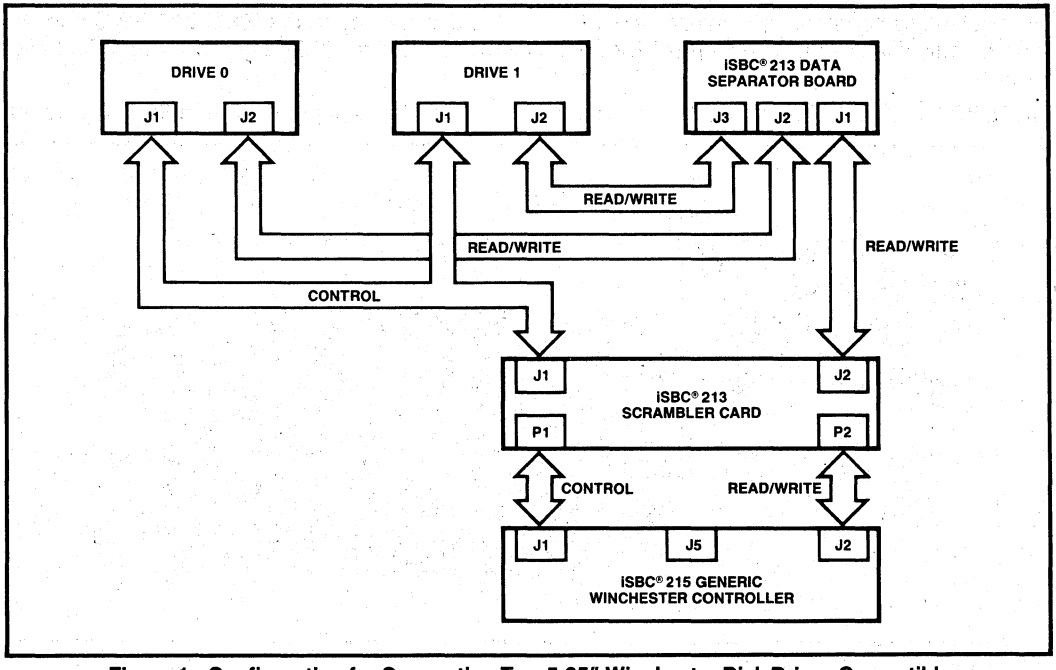


Figure 1. Configuration for Connecting Two 5.25″ Winchester Disk Drives Compatible With ST412 Interface

can correct an erroneous burst up to 11 bits in length.

iSBX™ Interface

Two iSBX bus connectors provide I/O expansion capability for the iSBC 215 GWC. With the optional addition of the iSBX 218A Flexible Disk Controller MULTIMODULE™ and or the iSBX 217C 1/4" Tape Interface Module, the iSBC 215 GWC can be configured into one of four types of peripheral subsystems, see Table 1.

Table 1. Peripheral Subsystem Configurations

	iSBC® 215G	iSBX™ 218A	iSBX™ 217C
Winchester Only	✓		
Winchester + Floppy	✓	✓	
Winchester + 1/4" Tape	✓		✓
Winchester + Floppy + 1/4" Tape	✓	✓	✓

Expanded I/O Capability

The iSBC 215 GWC allows the execution of user-written 8089 programs located in on-board or MULTIBUS system RAM. Thus the full capability of the 8089 I/O processor can be utilized for custom I/O requirements.

MULTIBUS® Interface

The iSBC 215 Generic Winchester Controller interfaces to the system CPU(s) through MULTIBUS memory. The iSBC 215 GWC directly addresses 16 megabytes of system memory. Commands are passed to and from the iSBC 215 GWC via memory based parameter blocks; these parameter blocks are executed directly by the iSBC 215 GWC thus off-loading the system CPU(s). Data transfers to and from the iSBC 215 GWC are done via the high speed DMA capability of the Intel 8089 I/O processor.

A block diagram of the iSBC 215 GWC is shown in Figure 2.

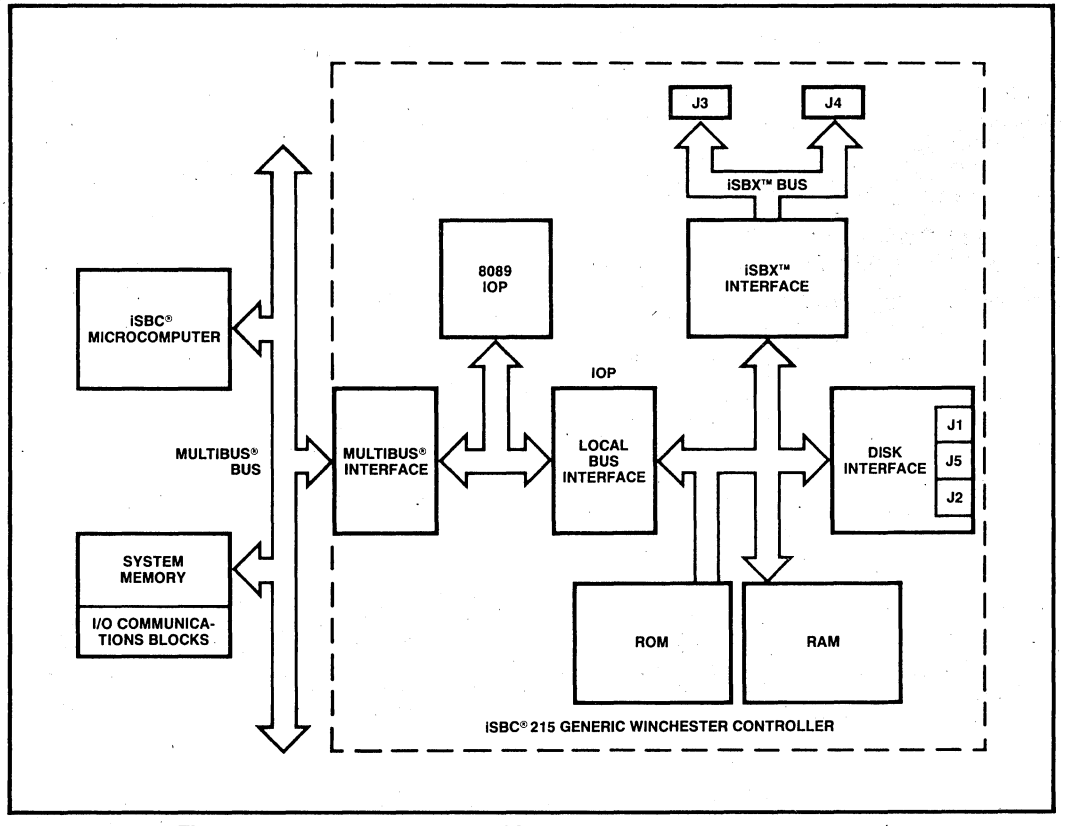


Figure 2. Block Diagram of iSBC® 215 Generic Winchester Controller

SPECIFICATIONS

Compatibility

CPU — Any MULTIBUS computer or system mainframe.

Disk Drives — 5.25″ Winchester disk drives compatible with the industry standard Seagate ST506/412 interface. Computer Memories Inc. (CMI) Model 5412 and 5619 drives have been extensively evaluated with the board and are known to function properly. Other drives should be extensively evaluated by the customer if they are to be used.

Transfer rate — 5 megabits per second.

Equipment Supplied

iSBC 215 Generic Winchester Controller

iSBC 213 Data Separator Board

iSBC 213 Scrambler Card

iSBC 213 Data Separator Kit Installation Guide

Reference Schematic

Scrambler to data separator and drive cabling and connectors are not supplied with the kit. Cables can be fabricated with flat cable and commercially available connectors as described in the iSBC 213 Installation Guide.

Physical Characteristics

iSBC® 215 GWC — Width: 6.75 in. (17.2 cm); Length: 12.0 in (30.5 cm); Height: 0.5 in. (1.3 cm)

Data Separator — Width: 5.50 in. (13.9 cm); Length: 7.25 in. (18.3 cm); Height: 0.5 in. (1.3 cm)

Scrambler — Width: 1.4 in. (3.6 cm); Length: 9.0 in. (22.9 cm); Height: 0.4 in. (1.0 cm)

Electrical Characteristics

iSBC® 215 GWC — +5 VDC @ 4.52A (max.); −5 VDC @ 0.015A (max.)[1]; +12 VDC @ 0.15A (max.)[2]; −12 VDC @ 0.055A (max.)[1,2]

iSBC® 213 Data Separator — +5 VDC @ 1.5A (max.)

NOTES:

1. On-board regulator and jumper allows −12 VDC usage from MULTIBUS.
2. Required for some iSBX modules.

Drives Per Controller

5.25″ Winchester Disk Drives — Up to two drives; compatible with the industry standard ST506/412 interface.

Floppy Disk Drives — Up to four drives via the optional iSBX 218A Floppy Disk Controller module connected to an iSBX connector on the iSBC 215 GWC board

1/4″ Tape Drives — Up to four drives via the optional iSBX 217C 1/4″ Tape Controller module connected to an iSBX connector on the iSBC 215 GWC board.

Environmental Characteristics

iSBC® 215 GENERIC WINCHESTER CONTROLLER

Temperature — 0°C to 55°C (operating) at 200 linear feet per minute (LFM) air velocity; −55°C to +85°C (non-operating)

Humidity — Up to 90% relative humidity without condensation (operating); all conditions without condensation or frost (non-operating)

iSBC 213 DATA SEPARATOR

Temperature — +10°C to 46°C (operating); −40°C to +62°C (non-operating)

Humidity — 8% to 80% (operating); 1% to 95% (non-operating)

Reference Manual

144780 — iSBC 215 Generic Winchester Controller Hardware Reference Manual (NOT SUPPLIED)

146232 — iSBC 213 Data Separator Kit Installation Guide (SUPPLIED)

Reference manuals may be ordered from any Intel sales representative, distributor office, or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, CA 95051.

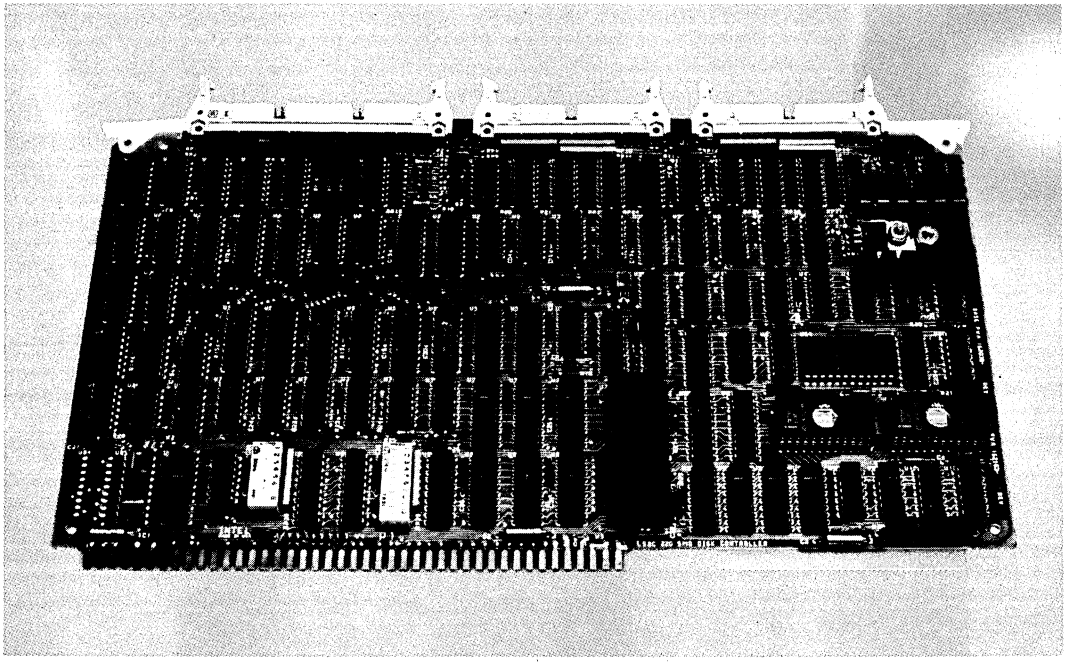
ORDERING INFORMATION

Part Number	Description
iSBC 215G Kit	5.25″ Winchester Controller Kit

iSBC® 220
SMD DISK CONTROLLER

- **Controls up to four soft sectored SMD interface compatible disk drives**
- **12 MB to 2.4 GB per controller**
- **Compatible with all iSBC® 80, iSBC® 88, and iSBC® 86 Single Board Computers**
- **Intel® 8089 I/O Processor provides two high speed DMA channels as well as controller intelligence**
- **Software drivers available for iRMX™ 86 and iRMX™ 88 operating systems**
- **On-board diagnostic and ECC**
- **Full sector buffering on-board**
- **Capable of addressing 1 MB of system memory**
- **SMD interface available on Winchester, CMD, SMD and large fixed-media drives**

The iSBC 220 SMD Disk Controller brings very large mass storage capabilities to any iSBC 80, iSBC 88, or iSBC 86 MULTIBUS system. The controller will interface to any soft sectored disk drive conforming to the industry standard SMD interface. Using simplified cable connections, up to four drives may be connected to the iSBC 220 Controller Board to give a total maximum capacity of 2.4 gigabytes. The Intel 8089 I/O Processor simplifies programming through the use of memory-based parameter blocks. A linked list technique allows the user to perform multiple disk operations.



AFN-01794A

OCTOBER 1984
ORDER NUMBER: 143283-002

FUNCTIONAL DESCRIPTION

Full On-Board Buffer

The iSBC 220 SMD Controller contains enough on-board RAM for one full sector buffering. The controller is designed to make use of this buffer in all transfers. The on-board sector buffer prevents data overrun errors and allows the iSBC 220 SMD Controller to occupy any priority slot on the MULTIBUS.

ECC

High data integrity is provided by on-board Error Checking Code (ECC) logic. When writing sector ID or data fields, a 32-bit Fire code, for burst error correction, is appended to the field by the controller. During a Read operation, the same logic regenerates the ECC polynomial and compares this second polynomial to the appended ECC. The ECC logic can detect an erroneous data burst up to 32 bits in length and using an 8089 alrogithm can correct an erroneous burst up to 11 bits in length.

SMD Interface

High speed, reliable data transfers are a major benefit of using the SMD interface. A data transfer rate of 1.2 MB is accomplished by using separate (radial) differential data line cabling for each drive. Control signals are daisy-chained from drive to drive.

Defective Track Handling

When a track is deemed defective, the host processor reformats the track, giving it a defective track code and enters the address of the next available alternate track. When the controller accesses a track previously marked defective, the controller automatically seeks to the assigned alternate track. The alternate track seek is totally automatic and invisible to the user.

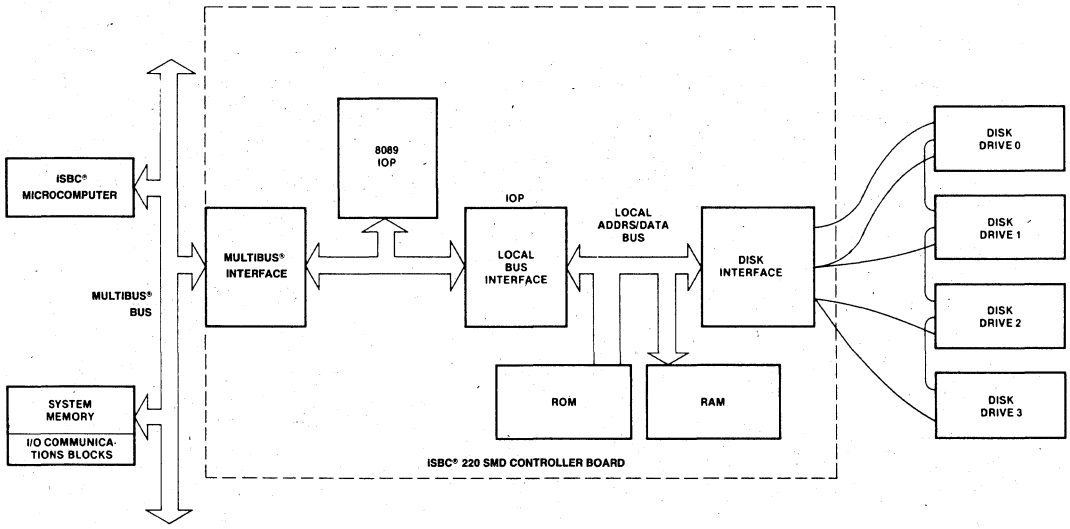


Figure 1. Simplified Block Diagram of iSBC® 220 SMD Disk Controller

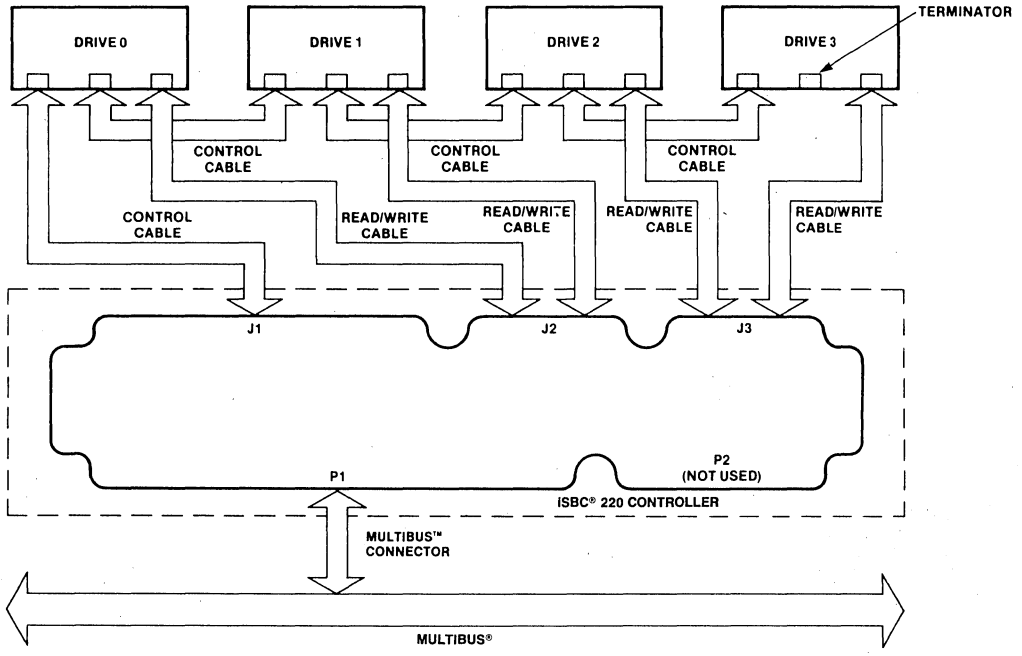


Figure 2. Typical Multiple Drive System

SPECIFICATIONS

Compatibility

CPU — Any iSBC MULTIBUS computer or system mainframe

Disk Drive – Any soft sectored SMD interface-compatible disk drive

Equipment Supplied

iSBC 220 SMD Disk Controller

Reference schematic

Controller-to-drive cabling and connectors are not supplied with the controller. Cables can be fabricated with flat cable and commercially-available connectors as described in the iSBC 220 SMD Disk Controller Hardware Reference Manual.

Physical Characteristics

Width — 6.75 in. (17.15 cm)

Height — 0.5 in. (1.27 cm)

Length — 12.0 in. (30.48 cm)

Shipping Weight — 19 oz (0.54 kg)

Mounting — Occupies one slot of iSBC system chassis or cardcage/backplane

Electrical Characteristics

Power Requirements

+ 5 VDC @ 3.25A max

– 5 VDC @ 0.75A max[1]

Note 1: On-board voltage regulator allows optional – 12 VDC usage from MULTIBUS.

Data Organization and Capacity

Bytes per Sector[2] — 128 256 521 1024

Sectors per Track[2] — 108 64 35 18

Note 2: Software selectable.

Table 1. Drive Characteristics (Typical)

Disk (spindle) Speed	3600 rpm
Tracks per Surface	823
Head Positioning	Closed loop servo type, track following
Access Time	Track to Track 6 ms Average 30 ms Maximum 55 ms
Data Transfer Rate	1.2 megabytes/second
Storage Capacity	12 to 2.4 gigabytes

Environmental Characteristics

Temperature — 0°C to 55°C (operating); −55°C to +85°C (non-operating)

Humidity — Up to 90% relative humidity without condensation (operating); all conditions without condensation or frost (non-operating)

Reference Manual

121597-001 — iSBC 220 SMD Disk Controller Hardware Reference Manual (NOT SUPPLIED)

Reference manuals may be ordered from any Intel sales representative, distributor office, or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, CA 95051.

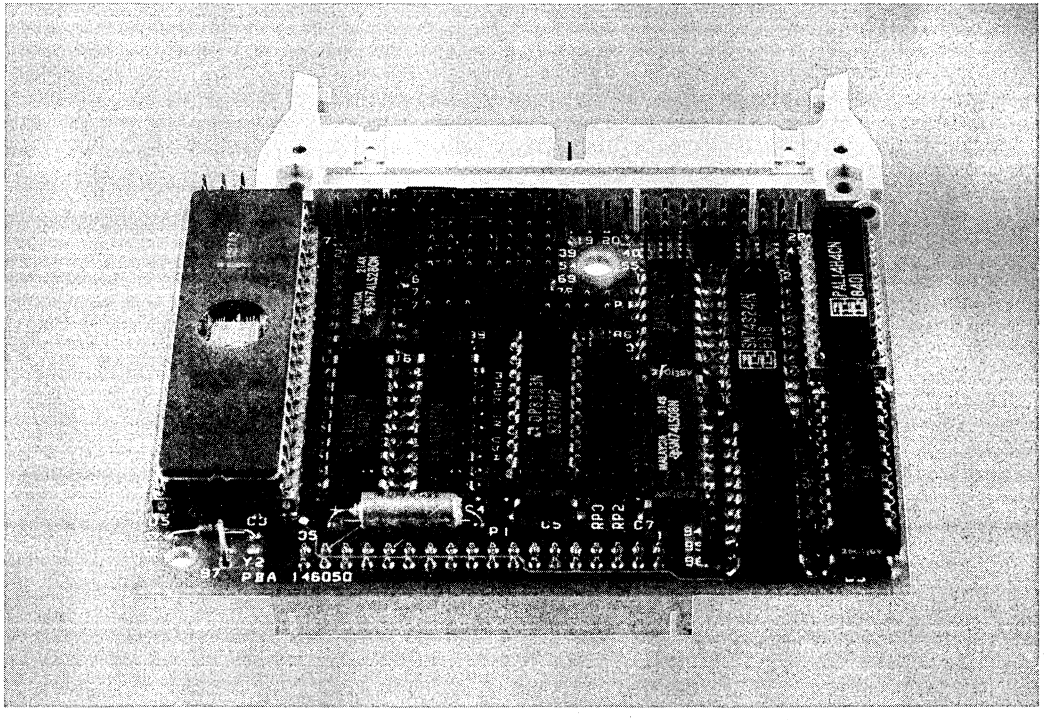
ORDERING INFORMATION

Part Number	Description
SBC 220	SMD Disk Controller

iSBX™ 217C ¼-INCH TAPE DRIVE INTERFACE MULTIMODULE™ BOARD

- **iSBX™ MULTIMODULE™ interface provides tape backup capability for iSBC® 215 Generic Winchester Controller**
- **Configurable to interface with up to four QIC-02 compatible or 3M HCD-75 compatible tape drives**
- **Implements the QIC-02 with parity streaming tape interface standard**
- **Supports transfer rates of 90K, 30K or 17K bytes per second depending on tape speed**
- **Supported by iRMX™ 86 and XENIX* Operating Systems when used on iSBC® 215 Generic Winchester Controller board**
- **+5 volt only operation**

The iSBX™ 217C ¼-Inch Tape Drive Interface module is a member of Intel's family of iSBX bus compatible MULTIMODULE™ products. iSBX MULTIMODULE boards plug directly onto any iSBX bus compatible host board, offering incremental on-board I/O expansion. The module is particularly useful for implementing cartridge tape back-up capability directly on the iSBC® 215 Generic Winchester Disk Controller via DMA. The iSBX 217C board can also provide a low-cost tape storage interface for any Intel single board computer, with an iSBX connector, via programmed I/O. The iSBX 217C module interfaces with up to four streaming tape drives. Typically, these drives provide 20 to 45 megabytes of storage each. When used in conjunction with these drives and the iSBC 215 board, the module can transfer 20 megabytes of data from disk to tape in about fourteen minutes. Alternatively, the iSBX 217C board can interface with up to four 3M Company HCD-75 compatible start/stop tape drives, for those applications requiring access to individual data files on tape.



* XENIX is a trademark of Microsoft Corporation.

June, 1984
Order Number: 210817-002

FUNCTIONAL DESCRIPTION

The iSBX 217C module implements an interface between a host iSBC board and a cartridge ¼-inch magnetic tape drive, with a minimum of host software overhead. Data transfers may occur in either a direct memory access (DMA) or programmed I/O mode. The DMA mode is available only with host iSBC boards which have DMA capability. In both modes, the host must be able to transfer data at a rate of 90K, 30K or 17K bytes per second, depending on the speed of the tape drive.

Communication with the iSBC® Host

A command plus one-to-five parameter bytes are issued by the host iSBC board to the iSBX 217C module to initiate any tape interface operation. Commands for the QIC-02 and 3M interfaces are summarized in Table 1. If the function is a Read or a Write operation, the host must then be ready to transfer data a byte at a time to or from the module. In programmed I/O mode, with QIC-02 drives, the host polls the iSBX 217C status port to learn when the tape interface is ready for the next 512 byte data block. During the data block transfer, the host is interrupted by MWAIT/ when the interface is ready to transfer a data byte. With 3M tape drives, the host may be interrupted or use MDRQT to detect when the module is ready for the next byte transfer. In DMA mode, the host board uses the DMA Request signal (MDRQT) of the iSBX bus to synchronize the data transfer. At the conclusion of a tape operation, the iSBC host must read one or more of the iSBX 217C module's Sense Bytes to receive status information on the completed operation. When the iSBX 217C module is used on the iSBC 215 Generic Winchester Controller board, these host requirements are fulfilled by the standard on-board firmware and are transparent to the user.

Table 1. Commands required by QIC-02 and 3M tape drives. Number indicates the parameter bytes required by the command. N indicates the command is not supported by the drive.

Hex Code	Command	Parameter Bytes		Type of Command
		QIC-2	3M	
00	RESET iSBX 217C BOARD	1	1	a
01	INITIALIZE DRIVE	1	1	a
02	WRITE A BLOCK	1	3	b
03	WRITE A FILE MARK	1	1	a
04	READ A BLOCK	1	3	b
05	READ FILE MARK COMMAND	1	N	a
06	READ STATUS	1	1	a
07	REWIND	1	N	a
08	RETENSION	1	N	a
09	ERASE TAPE	1	N	a
0C	UNLOAD TAPE	N	1	a
14	CONTINUE	N	1	a
15	WRITE RAM	N	5	b
16	READ RAM	N	5	b
17	VERIFY	N	5	a
18	RUN SELFTEST 1	1	N	a
1A	READ EXTENDED STATUS	1	N	a
1B	SET ALTERNATE SELECT MODE	1	N	a
1C	RETURN RAW DRIVE STATUS	1	N	a
20	RESET BAD PARITY FLAG	0	N	c
40	START OF TRANSFER (SOT)	1	1	c
80	END OF TRANSFER (EOT)	1	1	c
81	PAUSE COMMAND	1	N	c
82	RELEASE PAUSE COMMAND	1	N	c

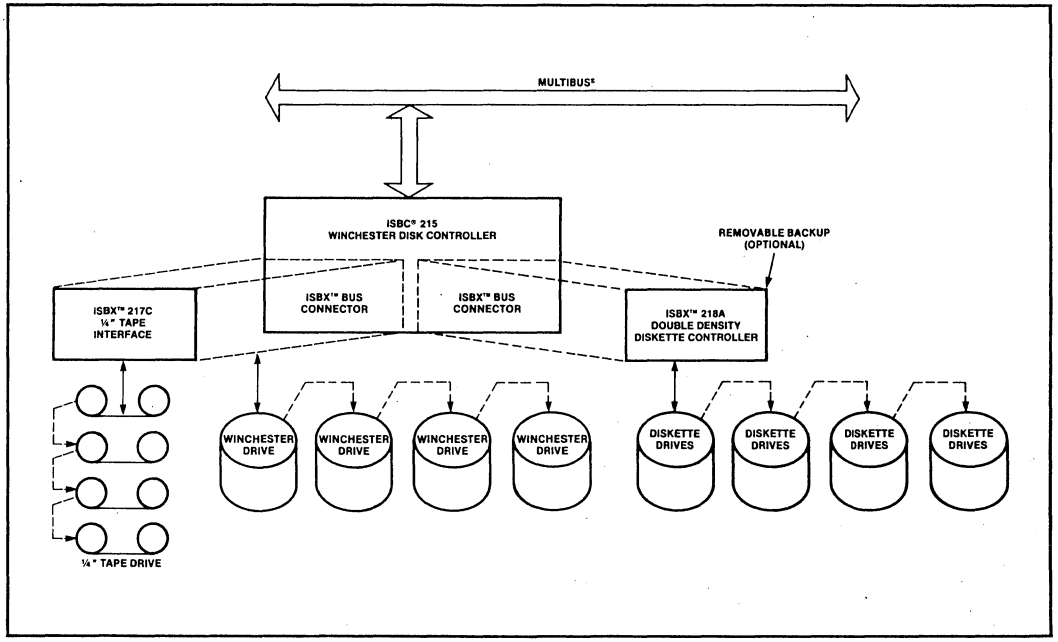


Figure 1. Subsystem Configuration (with optional Diskette and ¼" Tape Backup)

SPECIFICATIONS

Compatibility

Host — Any iSBC single board computer or peripheral controller with an iSBX connector. The iSBC 215 Generic Winchester Controller includes on-board firmware to support the iSBX 217C under either the iRMX 86 or XENIX Operating Systems. The firmware on the iSBC 215A and iSBX 215B Winchester Controllers cannot support the iSBX 217C module.

Drives — Any QIC-02 or 3M HCD-75 interface compatible cartridge ¼-inch magnetic tape drive.

Transfer Rate

90K (one byte every 11 microseconds), 30K (one byte every 33 microseconds) or 17K (one byte every 53 microseconds) depending on tape drive speed.

Equipment Supplied

iSBX 217C Interface Module
Reference Schematic

Controller-to-drive cabling and connectors are not supplied. Cables can be fabricated with flat cable and commercially-available connectors as described in the Hardware Reference Manual.

Nylon mounting bolts

Physical Characteristics

Width — 3.08 inches (7.82 cm)

Height — 0.809 inches (2.05 cm)

Length — 3.70 inches (9.40 cm)

Shipping Weight — 3.5 ounces (99.2 gm)

Mounting — Occupies one single-wide iSBC MULTIMODULE position on boards

Electrical Characteristics

Power Requirements — +5 VDC @ 1.5 A

Environmental Characteristics

Temperature — 0°C to 55°C (operating) @200 LFM; −55°C to +85°C (non operating)

Humidity — Up to 90% relative humidity without condensation (operating); all conditions without condensation or frost (non-operating)

Reference Manual

144260-001 — iSBX 217C Board Hardware Reference Manual (NOT SUPPLIED)

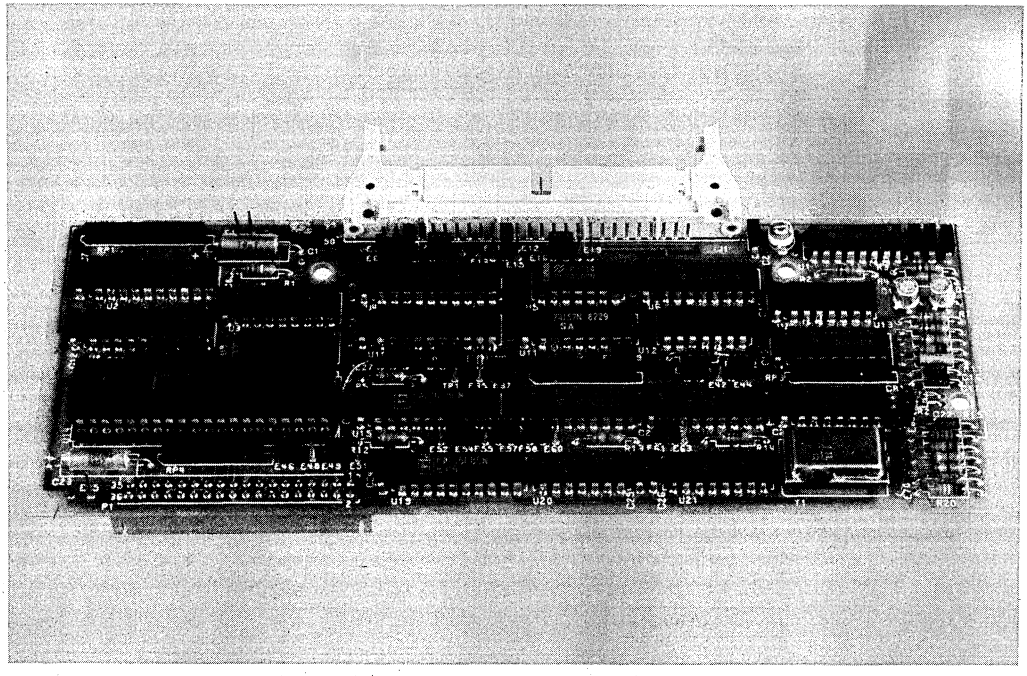
ORDERING INFORMATION

Part Number	Description
SBX 217C	Cartridge ¼-Inch Tape Drive Interface

iSBX™ 218A
FLEXIBLE DISK CONTROLLER

- **iSBX™ bus compatible 8" or 5.25" floppy diskette controller module**
- **Hardware and software compatible with iSBX™ 218 module**
- **Controls most single/double density and single/double sided floppy drives**
- **User programmable drive parameters allow wide choice of drives**
- **Motor on/off latch under program control**
- **Drive-ready timeout circuit for 5.25 inch floppy drives**
- **Phase lock loop data separator assures data integrity**
- **Read and write on single or multiple sectors**
- **Single +5 volt supply required**

The Intel iSBX™ 218A Flexible Disk Controller module is a software and hardware compatible replacement for the iSBX 218 module and provides additional features. The iSBX 218A module is a double-wide iSBX module floppy disk controller capable of supporting virtually any soft-sectored, single/double density and single/double sided floppy drives. The controller can control up to four drives. In addition to the standard IBM 3740 and IBM system 34 formats, the controller supports sector lengths up to 8192 bytes. The iSBX 218A module's wide range of drive compatibility is achieved without compromising performance. The operating characteristics are specified under user control. The controller can read and write either single or multiple sectors.



OCTOBER 1984
ORDER NUMBER: 503810-002

FUNCTIONAL DESCRIPTION

Intel's 8272 floppy Disk Controller (FDC) chip is the heart of the iSBX 218A Controller. On-board data separation logic performs standard MFM (double density) and FM (single density) encoding and decoding, eliminating the need for external separation circuitry at the drive. Data transfers between the controller and memory are managed by the intelligent device on the host board (usually an Intel 8-bit or 16-bit CPU). A block diagram of the iSBX 218A Controller is shown in Figure 1.

Universal Drive and iSBX™ 218A Controller

Because the iSBX 218A Controller has universal drive compatibility, it can be used to control virtually any standard- or mini-sized diskette drive. Moreover, the iSBX 218A Controller fully supports the iSBX bus and can be used with any single board computer which provides this bus interface. Because the iSBX 218A Controller is programmable, its performance is not compromised by its universal drive compatibility. The track-to-track access, head-load, and head-unload characteristics of the selected drive model are program specified. Data may be organized in sectors up to 8192 bytes in length.

Interface Characteristics

The standard iSBX 218A Controller includes an Intel 8272 Floppy Disk Controller chip which supports up to four drives, single or double sided.

SIMPLIFIED INTERFACE — The cable between the iSBX 218A Controller and the drive(s) may be low cost, flat ribbon cable with mass teminations connectors. The mechanical interface to the

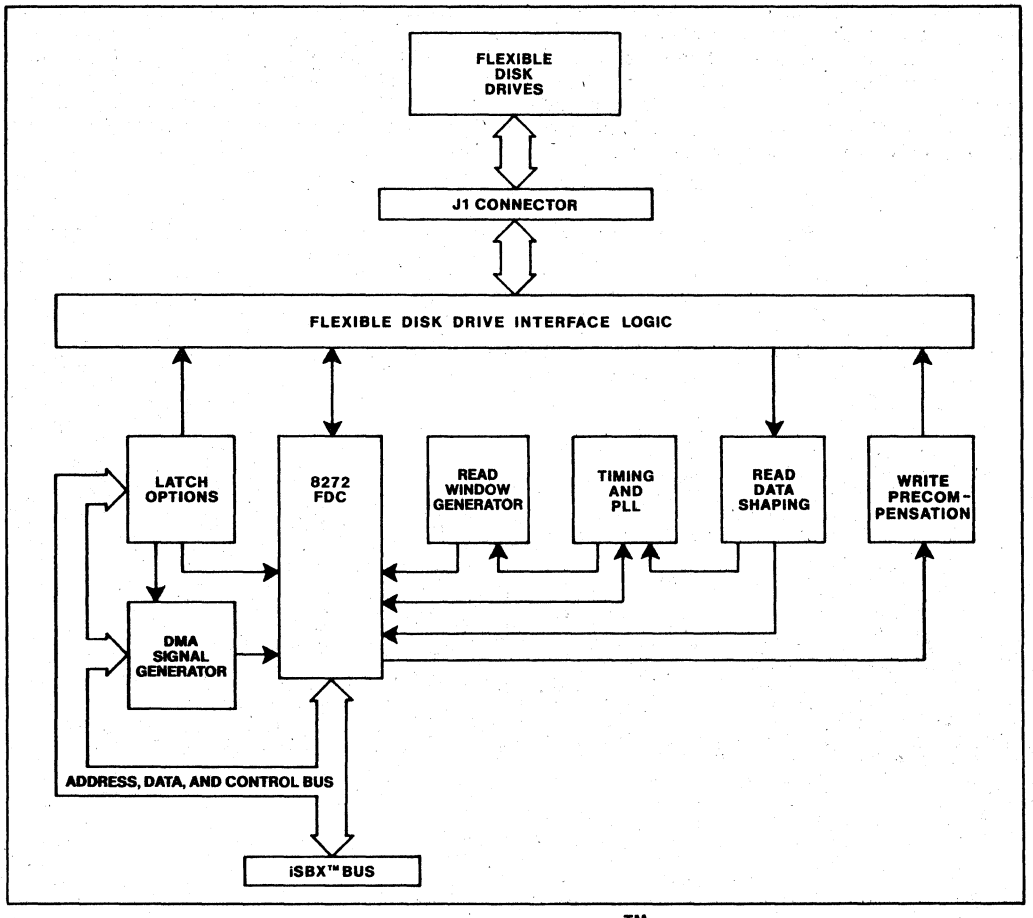


Figure 1. Block Diagram of iSBX™ 218A Board

board is a right-angle header with locking tabs for security of connection.

PROGRAMMING – The powerful 8272 FDC circuit is capable of executing high-level commands that simplify system software development.The device can read and write both single and multiple sectors. CRC characters are generated and checked automatically. Recording density is selected at each Read and Write to support the industry standard technique of recording basic media information on Track 0 of Side 0 in single density, and then switching to double density (if necessary) for operations on other tracks.

PROGRAM INITIATION – All diskette operations are initiated by standard iSBX bus input/output (I/O) operations through the host board. System software first initializes the controller with the operating characteristics of the selected drive. The diskette is then formatted under program control. Data transfers occur in response to commands output by the CPU.

DATA TRANSFER – Once a diskette transfer operation has been initated, the controller will require a data transfer every 13 microseconds (double density) or 26 microseconds (single density). Most CPUs will operate in a polled mode, checking controller status and transferring bytes when the controller is ready. Boards utilizing the intel 8080 chip, such as the iSBC 80/10B board, will be restricted to single density operation with the iSBX 218A Controller, due to these speed requirements.

DMA OPERATION – The iSBX 218A module can be used either with or without a DMA controller on the host board. Standard DMA controllers provide a DACK (DMA Acknowledge) signal for proper DMA operation with the 8272. The iSBX 218A's on-board DACK generator provides the interface to allow the iSBX 218A module to be used with DMA controllers such as Intel's 8089 and 80186 processors that do not provide a DACK signal.

SPECIFICATIONS

Compatibility

CPU – Any single board computer or I/O board implementing the iSBX bus interface and connector.

Devices – Double or single density standard (8") and mini (5¼") flexible disk drives. The drives may be single or double sided. Drives known to be compatible are indicated in the table to the right.

Standard (8")		Mini (5¼")	
Caldisk	143M	Shugart	450/400
Remex	RFD 4000	Shugart	460/410
Memorex	550	Micropolis	1015-IV
MFE	700	Pertec	250
Siemens	FDD 200-8	Siemens	200-5
Shugart	SA 850/800	Tandon	TM-100
Shugart	SA 860/810	CDC	9409
Pertec	FD650	MPI	51/52/91/92
CDC	9406-3		

Data Organization and Capacity

Standard Size Drives

	Double Density						Single Density					
	IBM System 34			Non-IBM			IBM System 3740			Non-IBM		
Bytes per Sector	256	512	1024	2048	4096	8192	128	256	512	1024	2048	4096
Sectors per Track	26	15	8	4	2	1	26	15	8	4	2	1
Tracks per Diskette	77			77			77			77		
Bytes per Diskette (Formatted, per diskette surface)	512,512 (256 bytes/sector) 591,360 (512 bytes/sector) 630,784 (1024 bytes/sector)			630,784			256,256 (128 byte/sector) 295,680 (256 bytes/sector) 315,392 (512 bytes/sector)			315,392		

Diskette – Unformatted IBM Diskette 1 (or equivalent single-sided media); unformatted IBM Diskette 2D (or equivalent double-sided).

Equipment Supplied

iSBX 218A Controller
Reference Schematic
Controller-to-drive cabling and connectors are not supplied with the controller. Cables can be fabricated with flat cable and commercially-available connectors as described in the iSBX 218A Hardware Reference Manual.
Nylon Mounting Screws and Spacers

Physical Characteristics

Width – 3.15 inches (8.0 cm)
Height – 0.83 inches (2.1 cm)
Length – 7.5 inches (19.1 cm)
Weight – 4.5 ounces (126 gm)
Mounting – Occupies one double-wide iSBX MULTIMODULE™ position on boards; increases board height (host plus iSBX board) to 1.13 inches (2.87 cm).

Electrical Characteristics

Power Requirements – +5VDC @ 1.7A max.

Environmental Characteristics

Temperature – 0°C to +55°C (operating); −55°C to +85°C (non-operating).
Humidity – Up to 90% Relative Humidity without condensation (operating); all conditions without condensation or frost (non-operating).

Reference Manual

145911-001 – iSBX 218A Flexible Disk Controller Hardware Reference Manual (NOT SUPPLIED).

Reference manuals may be ordered from any Intel sales representative, distributor office, or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, CA 95051.

Drive Characteristics

	Standard Size	Mini Size
	Double/Single Density	Double/Single Density
Transfer Rate (K bytes/sec)	62.5/31.25	31.25/15.63
Disk Speed (RPM)	360	300
Step Rate Time (Programmable)	1 to 16 msec/track in 1 msec increments	2 to 32 msec/track in 2 msec increments
Head Load Time (Programmable)	2 to 254 msec in 2 msec increments	4 to 508 msec in 4 msec increments
Head Unload Time (Programmable)	16 to 240 msec in 16 msec increments	32 to 480 msec in 32 msec increments

ORDERING INFORMATION

Part Number	Description
SBX 218A	Flexible Disk Controller

Human Interface

9

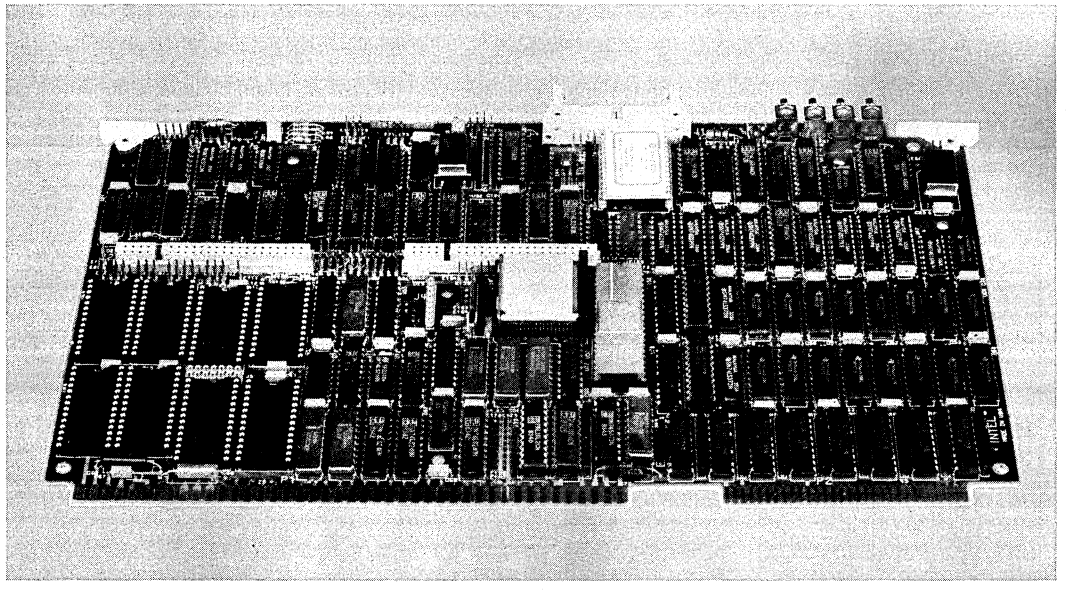
PRELIMINARY

iSBC™ 186/78A
INTELLIGENT VIDEO GRAPHICS SUBSYSTEM

- **iAPX 186 Integrated Microprocessor with 8 MHz CPU clock**
- **Includes Intel's 82720 Graphic Display Controller**
- **Programmable 50 or 60 Hz frame rate**
- **Maximum resolution of 1024 x 768 x 4 (interlaced) or 640 x 480 x 4 (non-interlaced)**
- **Drawing rate of 150K pixels/second**
- **Multi-write into all planes simultaneously**
- **Look up table allows 16 of possible 4096 colors**
- **Interfaces to either monochrome or color raster-scan display monitors**
- **Eight Universal Memory Sites for local RAM or ROM store**
- **MULTIMODULE™ expansion provided via two iSBX™ bus connectors**
- **Full RS343 or RS170 support**
- **DMA to local bus from iSBX™ and MULTIBUS®**

The iSBC® 186/78A Video Graphics Subsystem is the newest member of Intel's growing family of microcomputer graphics products. It was designed to provide an economical, off-the-shelf solution for OEM applications. A powerful microprocessor, the 80186, is dedicated to managing the board functions, thereby off-loading the central CPU board in the host system. The sub-system clock, local memory sockets, MULTIBUS® interface logic, and priority interrupt logic and programmable timers all reside on the board.

In addition to the local microprocessor, powerful bit-mapped graphics are made possible by the Intel 82720. This allows support of high level drawing commands including arcs, circles, rectangles, area filling, zoom, panning and scrolling. Applications that will benefit from the utility of the iSBC 186/78A subsystem are engineering workstations, process control monitoring, automatic test and instrumentation.



November, 1984
Order Number: 231035-002

FUNCTIONAL DESCRIPTION

Graphics Subsystem

The iSBC 186/78A subsystem integrates both a high performance programmable microprocessor and medium resolution graphics capability onto one board, serving both the computational and display requirements dictated by the application.

Such high integration results in two significant benefits to the user: (1) increased system performance by offloading the host CPU board from graphics routines, and (2) increased savings due to the compact, single board implementation. The iSBC 186/78A subsystem is capable of running Intel's Graphics Standard Software in addition to user-specific application programs. By providing such decentralized graphics utility, a multi-user system may now be configured more easily and at a cost that is more directly proportionate to the number of users serviced.

Architecture

A more traditional approach to graphics expansion would have yielded two functional blocks on a single iSBC board; e.g., graphics control and MULTIBUS® interface logic. However, Intel has integrated a third block, the dedicated 80186 and its peripheral circuitry, to act as a "control center" for the local graphics platform. Each functional area is highlighted in Figure 1, and detailed in separate sections.

SUBSYSTEM CONTROL CENTER

Central Processing Unit

The 80186 component is a high-performance, high-integration 16-bit microprocessor. It combines several of the most common system components onto a single chip (i.e. Direct Memory Access, Interval Timers, Clock Generator and Programmable Interrupt Controller) and provides a 30% performance improvement over the 8086 at an equivalent clock rate.

Three internal 16-bit programmable timers are provided. Two of these are highly flexible and are connected to four external pins on the iSBC 186/78A subsystem (two per timer). They can be used to count or time external events, generate nonrepetitive waveforms, etc. The third timer is not connected to any external pins, and is useful for real-time coding and time delay applications. In addition, this third timer can be used as a prescaler for the other two, or as a DMA request source.

Both DMA channels provided by the 80186 are supported which allows a direct path from the MULTIBUS or iSBX bus to local memory. Indirect access to the display memory is also possible under GDC control.

The system software can configure each timer independently to select the desired function. Available functions include: interrupt on terminal count, programmable one-shot, rate generator, square-wave generator, software triggered strobe, hardware triggered strobe and event counter. The contents of each counter may also be read at any time during system operation.

Instruction Set

The 80186 instruction library is a superset of that for the 8086. Therefore, object code compatibility was maintained while 10 instructions were added. The new instructions include: Block I/O, Enter and Leave subroutines, Push Immediate, Multiply Quick, Array Bounds Checking, Shift and Rotate by Immediate, and Pop and Push All.

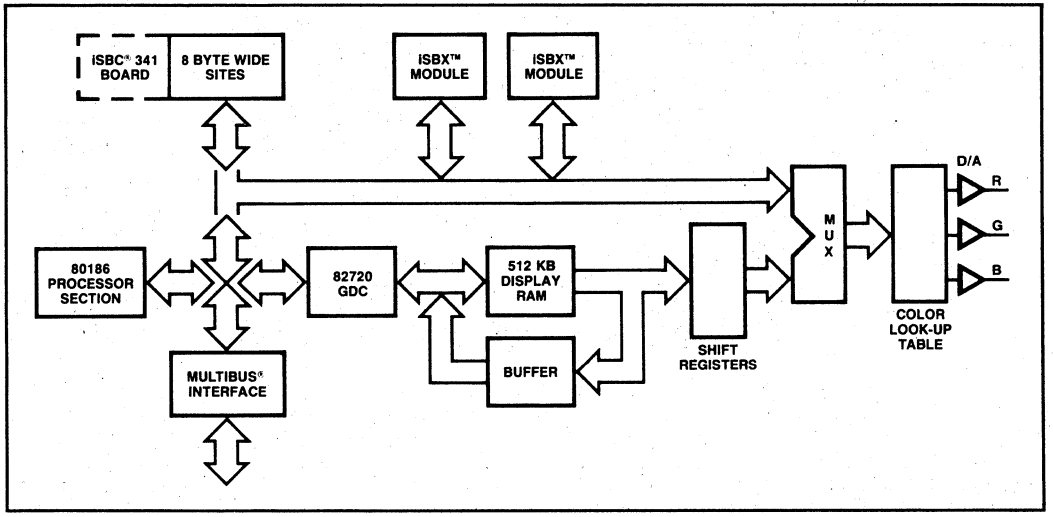


Figure 1. Block Diagram

Architectural Features

A six-byte instruction queue provides prefetching of sequential instructions and can reduce the 500 nsec minimum instruction cycle to 333 nsec for queued instructions. The stack oriented architecture readily supports modular programming by facilitating fast, simple intermodule communication along with other programming constructs needed for asynchronous real-time systems. Using a windowing technique and external logic, the full 16 MByte addressing range of the IEEE-796 MULTIBUS Standard is available to the user.

This dynamic relocation scheme allows ease in segmentation of pure procedure and data for efficient memory utilization. Four segment registers (code, stack, data, extra) contain program loaded offset values which are used to map 16-bit addresses to 20-bit addresses. Each register maps 64 KBytes at a time and activation of a specific register is controlled, both explicitly by program control, and implicitly by specific functions and instructions. A flag byte signaling mechanism aids in creating an interprocessor communication scheme. This includes: (1) the ability to set/reset interrupts with MULTIBUS commands and (2) board reset.

GRAPHICS PROCESSOR FUNCTIONALITY

Graphics Display Controller

The Intel 82720 GDC is an intelligent graphics controller designed to operate as the heart of a raster-scan computer graphics display system. The 82720 performs all the basic timing needed to generate the raster display and manage the display memory. In addition, the 82720 supports several high level graphics figure drawing functions. Table 1 highlights the 82720 command set.

The 186/78A uses the graphics mode of the 82720 GDC although the subsystem does not provide an external character generator. Also, there is no external zoom circuitry, hence, only the zoom-write feature is supported. DMA is supported, as well as all of the other GDC features. The pixel clock rate depends on the desired resolution and the number of colors. The clock rate of 25 MHz is jumper selectable at 20 MHz for 640 x 480 x 16 colors.

Graphics Standard Software (OPTIONAL)

Intel also produces a software driver package that will interface the iSBC 186/78A subsystem to one of Graphics Software Standards defined by the ANSI X3 organization. Please consult your Intel representative for details.

CRT INTERFACE

The iSBC 186/78A subsystem will interface to many monochrome and RGB (Red, Green and Blue) color display monitors. For monochrome monitors, TTL and analog level signals for video, vertical sync and horizontal sync, or combined sync are provided. When operating in the color mode, the iSBC 186/78A subsystem will output analog-level Red, Green and Blue video, with a maximum of 16 different colors displayed. Either RS 170 or RS 343 may be supported under user-programmable configuration.

Table 1. 82720 Command Library

Video Control Commands	
RESET:	Resets the GDC to its idle state.
SYNCH:	Specifies the video display format.
Display Control Commands	
START:	Ends idle mode and unblanks the display.
BCTRL:	Controls the blanking and unblanking of the display.
ZOOM:	Specifies the zoom factors for graphics character writing.
CURS:	Sets the position of the cursor in display memory.
PRAM:	Defines the starting address and lengths of display areas, and specifies the eight bytes for the graphics character.
PITCH:	Specifies the width of the X dimension in display memory.
Drawing Control Commands	
WDAT:	Writes data words or bytes into display memory.
MASK:	Sets the mask register contents.
FIGS:	Specifies the parameters for the drawing processor.
FIGD:	Draws the figure as specified.
GCHRD:	Draws the graphics character into display memory.
Data Read Commands	
RDAT:	Reads data words or bytes from display memory.
CURD:	Reads the cursor position.
LPRD:	Reads the light pen address.

The analog signal originates in a sophisticated look-up table that allows for a possible 4096 colors and contains the three DACs required. The Red and Blue output signals are always analog, between 0 and 0.6 volts. The green can be analog or TTL compatible. When it is analog, the synch signals may be combined with Green output. Hsynch and Vsynch may be combined or separate.

MEMORY FUNCTIONALITY

Universal Memory Sites for Local Memory

Eight 28-pin JEDED-compatible sockets are provided for using 2732, 2764, 27128, 27256 and 27512 EPROMs and their respective ROMs. When using the 27256s, the on-board EPROM capacity is 256 KBytes. Other JEDEC standard pinout devices are also supported, including byte-wide static RAMs and iRAMs. Further expansion to a total of 12 sockets is provided by adding the iSBC 341 board giving a total of 384 Kbytes of storage using 27256s or 768 Kbytes using 27512s.

The eight sockets are divided into four blocks of two each (for high and low byte), or six blocks when using the iSBC 341 board. This allows mixing many different kinds of 28-pin devices for increased application flexibility. All devices on the expansion module must be the same and only two different components may be used at any one time, however. The memory decode PAL is socketed so that the user may custom configure to suit unique situations.

Display Memory

The iSBC 186/78A subsystem contains 512 Kbytes of high speed display memory, all of which is under the control of the 82720. The 82720 manages both writing and reading data to and from the screen and refreshing the screen.

The configuration of on-board display memory may be set under user program control which is read or written, 16-bits at a time, by the 82720. When displaying, the 82720 starts at the top left hand corner of the screen and sequences down the screen toward the bottom right hand corner. There are two types of memory cycles; display cycles, and read-modify-write cycles. During display cycles, data is read from the display memory and sent to the CRT for display. During RMW cycles, data is transferred between the GDC and the display memory. This transfer either writes data into the display memory or reads data from the display memory by the GDC.

In monochrome mode, all 256K, 16-bit words are treated as a contiguous block of memory, where a logical "1" in memory is displayed as an illuminated pixel. In the color mode, four color planes exist in memory and are written into (multi-write) and displayed simultaneously. Each plane consists of 16K words, each with 16 bits per pixel.

INTERRUPT CONTROL

The iSBC 186/78A board makes available the programmable interrupt controller (PIC) in the 80186 component, and allows 5 on-board vectored interrupt levels, including non-maskable interrupt (NMI). The highest priority interrupt is the Non-Maskable Interrupt (NMI) line which is tied directly to the 80186 CPU. This interrupt is typically used to signal catastrophic events (e.g. power failure). The PIC provides prioritization and vectoring for the other 4 interrupt requests from on-board I/O resouces and from the MULTIBUS system bus. The PIC then resolves the requests according to the programmable priority resolution mode, and if appropriate, issues an interrupt to the CPU.

Interrupt service requests to the iSBC 186/78A subsystem may originate from 22 sources. Table 2 contains a list of devices and functions capable of generating interrupts. Most of these interrupts are jumper configurable to the desired interrupt request level.

MULTIBUS® SYSTEM ARCHITECTURE

System Bus — Overview

The MULTIBUS system bus is Intel's industry standard (IEEE-796) microcomputer bus structure. Both 8- and 16-bit single board computers are supported with 24 address and 16 data lines. In its simplest application, the system bus allows expansion of functions already contained on the iSBC 186/78A subsystem (e.g., memory and digital I/O). However, the structure also allows very powerful distributed processing configurations with multiple processors and intelligent slave,

Table 2. Interrupt Request Sources

DEVICE	FUNCTION	NUMBER OF INTERRUPTS
MULTIBUS® Interface INT0-INT7	Requests from MULTIBUS® Resident Peripherals or Other CPU	8
Internal 80186 Timer And DMA	Timer 0, 1, 2, Outputs (Function Determined By Timer Mode) And 2 DMA Channel Interrupts	5
iSBX™ Connectors	Function Determined By iSBX™ MULTIMODULE™ Board	6
Bus Fail-Safe Timer	Indicates Addressed MULTIBUS® Resident Device Has Not Responded To Command Within 6 msec	1
GDC Vertical Retrace	Synchronization Of Monitor	1
Flag Byte	Board Identification	1

I/O and peripheral boards capable of solving the most demanding microcomputer applications. The MULTIBUS system bus is supported with a broad array of board level products, LSI interface components, along with detailed specifications and application notes.

Multimaster Capabilities

For those applications requiring additional processing capacity and the benefits of multiprocessing (i.e., several CPUs and/or controllers logically sharing system tasks through communication of the system bus), the iSBC 186/78A subsystem provides full MULTIBUS arbitration control logic. This control logic allows up to three iSBC 186/78A subsystems or other bus masters, including iSBC 80 family MULTIBUS compatible 8-bit single board computers, to share the system bus using a serial (daisy chain) priority scheme. A maximum of 16 bus-masters may share the MULTIBUS system bus with an external parallel priority decoder. In addition to the multiprocessing configurations made possible with multimaster capability, it also provides a very efficient mechanism for all forms of DMA (Direct Memory Access) transfers.

MULTIBUS® Expansion

Memory and I/O capacity may be expanded further and additional functions added using Intel MULTIBUS compatible expansion boards. Memory may be expanded by adding user specified combinations of RAM boards, EPROM boards, or memory combination boards. Input/Output capacity may be added with digital I/O and analog I/O expansion Boards. Mass storage capability may be achieved by adding single- or double-density diskette controllers, or hard disk controllers either through the use of MULTIBUS expansion boards or iSBX MULTIMODULE™ boards. Modular expandable backplanes and cardcages are available to support multi-board systems.

iSBX™ MULTIMODULE™ Expansion

The two 8/16-bit iSBX MULTIMODULE connectors allow the addition of I/O functionality to tailor the iSBC 186/78A board to the unique application requirements. The iSBX MULTIMODULE boards optimally support functions provided by VLSI peripheral components such as additional parallel and serial I/O, analog I/O, mass storage device control, and additional graphics power. By mounting directly on the single board computer, less interface logic, less power, simpler packaging, higher performance, and lower cost result when compared to other alternatives such as full-size iSBC boards.

Each of the iSBX connectors on the iSBC 186/78A subsystem provides all signals necessary to interface to the local on-board bus, including 16 data lines for maximum data transfer rates. All iSBX MULTIMODULE boards, designed with 8-bit data paths and using the 8-bit iSBX connector, are also supported on the iSBC 186/78A subsystem. A broad range of iSBX MULTIMODULE options are available from Intel. Custom iSBX modules may also be designed by using the iSBX bus interface specification and iSBX connectors, both available from Intel.

SOFTWARE SUPPORT

Run-time Support

Intel offers two run-time support packages for use on the iSBC 186/78A Video Graphics Subsystem: iRMX™ 88 Real-Time Multitasking Executive (Release 3) and the iRMX 86 Operating System (Release 6). The iRMX 88 executive is a simple, highly configurable and efficient foundation for small, high-performance applications. Its multitasking structure establishes a solid foundation for modular system design and provides task scheduling and management, intertask communication and synchronization, and interrupt servicing for a variety of peripheral devices. Other configurable options include terminal handlers, disk file system, debuggers and other utilities.

The iRMX 86 Operating System is a highly functional operating system with a very rich set of features and options based on an object-oriented architecture. In addition to being modular and configurable, functions beyond the nucleus include a sophisticated file management and I/O system, and a powerful human interface.

Development Environmment

Intel offers a family of tools to aid in the development of iSBC 186/78A subsystem based applications. These include full development systems, in-circuit emulators and programming languages. Some of the features of each are described below. Additional information regarding the development environment is available from your Intel representative.

The development cycle of iSBC 186/78-based products can be significantly simplified by using either a System 86/3XX or Intellec® Series Microcomputer Development System. The Assembler, Locating Linker, Library Manager, Text Editor and System Monitor are all supported by the ISIS-II disk-based operating system.

The Integrated Instrumentation In-Circuit Emulator (I^2ICE)™ provides the necessary link between an Intellec Development System and the "target" iSBC 186/78A execution system. In addition to providing the mechanism for loading executable code and data into the iSBC 186/78A boards, the I^2ICE 186 provides a sophisticated command set to assist in debugging software and final integration of the user hardware and software.

Intel has two systems implementation languages, PL/M-86 and C-86. Both are available under the iRMX 86 Operating System, in the System 86/3XX and the

Intellec Microcomputer Development System. PL/M-86 provides the capability to program in algorithmic language and eliminates the need to manage register usage or allocate memory while still allowing explicit control of the system's resources when needed. C-86 is especially appropriate in applications requiring portability and code density. FORTRAN 86, PASCAL 86, and BASIC 86 are also available under the iRMX 86 Operating System, in the System 86/3XX and the Intellec development system.

SPECIFICATIONS

WORD SIZE

Instruction — 8, 16, 24, or 32 bits
Data — 8 or 16 bits

SYSTEM CLOCK

8.00 Mhz ± 0.1%

INSTRUCTION CYCLE TIME

8 Mhz — 500 ns
— 333 ns (assumes instruction in queue)

Note: Basic instruction is defined as the fastest instruction time (i.e., two clock cycles.)

MEMORY RESPONSE TIME

286 ns for zero wait-states (address to data-valid)

MEMORY CAPACITY

EPROM	512 Kbytes (768K with iSBC 341 board) using 27512's
E²PROM	16 Kbytes (24K with iSBC 341 board) using 2817A's
iRAM	64 Kbytes (96K with iSBC 341 board) using 2186's
Static RAM	same as iRAM

CONNECTORS

Interface	Double-sided	Centers	Supplier
MULTIBUS® System	86 pin (P1)	0.156 in	Viking 3KH43/ 9AMK12 Wire Wrap
	60 (P2)	0.1	Viking 3KH30/ 9JNK
iSBX™ Bus, 8/16 Bit	38	0.1	iSBX™ 960-5
Video Interface			
1 pc.	26	0.1	Flat Cable Connector Type 3M 3399-6026
— or —			
4 pc's.	SMC	N/A	Sealectro Part No. 50-007-0000, for Belden 174/U Coax

PHYSICAL CHARACTERISTICS

Length — 12.00 in. (30.48 cm)

Height — 7.05 in. (17.90 cm)

Depth — 0.70 in. (1.78 cm)
— 1.13 in. (2.82 cm) with iSBC Memory Expansion, or iSBX MULTIMODULE boards.

Weight — 18.3 ounces (519 gm) excluding any MULTIMODULE boards.

ELECTRICAL CHARACTERISTICS

Power Requirements — 8.4A @ +5 ± 5% Vdc (Maximum); 4.9A @ +5 ± 5% Vdc (Typical)

ENVIRONMENTAL REQUIREMENTS

Operating Temperature — 0° to 55°C with 200 Linear feet per minute air flow

Relative Humidity — to 90% without condensation

REFERENCE MANUAL

147393-001 — iSBC 186/78A Video Graphics Subsystem Hardware Reference Manual

RELATED LITERATURE

210883-001 — MULTIBUS® Handbook
28001-001-iPLP720—Data Sheet (NAPLPS Interpreter)
280002-001-iVDI720—Data Sheet (Virtual Device Interface)

142686-001 — iSBX™ Specification

210451-001 — iAPX 80186 Data Sheet

210655-001 — Intel 82720 Data Sheet

Literature and Hardware Reference Manual may be ordered from an Intel Sales Representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

ORDERING INFORMATION

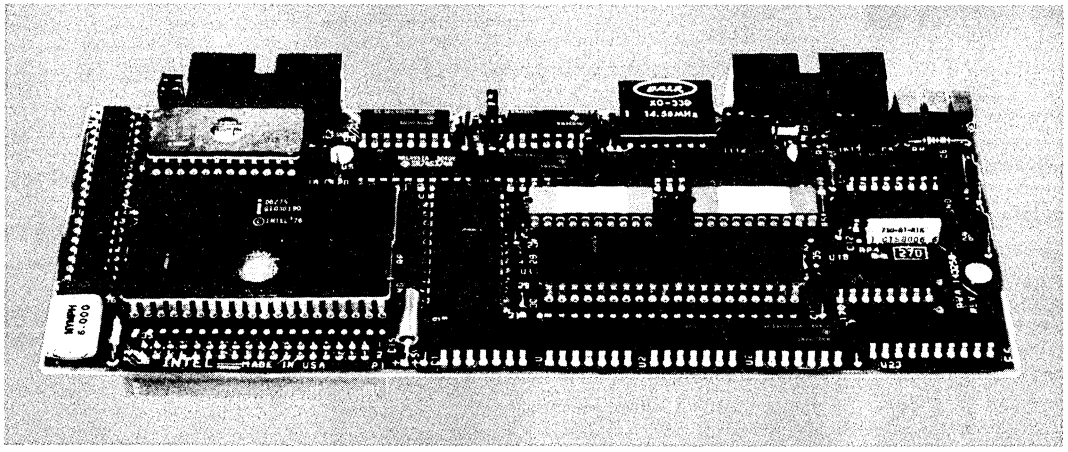
Part Number	Description
iSBC® 186/78A	Video Graphics Subsystem

iSBX™ 270
ALPHA-NUMERIC DISPLAY CONTROLLER

- **Complete video display controller on a double-wide iSBX™ MULTIMODULE™ board**
- **Interfaces to either black and white or color display monitors**
- **Displays 7 × 9, 5 × 7 or 6 × 8 character fonts**
- **High level software interface via a pre-programmed 8041A UPI**
- **Interchangeable character fonts available in EPROM**
- **Keyboard and light pen interface provided on-board**
- **50 Hz or 60 Hz frame rate operation**
- **Provides cursor control, reverse video, blinking, underline, highlight and page or scroll mode**
- **Compatible with all 8/16 bit iSBC® boards which support the Intel iSBX™ bus**
- **Graphics capability via pre-defined graphic character fonts**

The iSBX 270 Video Display Controller (VDC) is a complete video controller on a standard double wide Intel iSBX MULTIMODULE board. Providing either black and white (B&W) or eight-color displays, the iSBX 270 VDC brings alphanumeric video control to the iSBX bus. Any computer board or system supporting the Intel iSBX MULTIMODULE bus is compatible with the iSBX 270 VDC, including most board and system products from Intel. Additionally, the iSBX 270 VDC supports keyboard and light pen I/O on-board; this simplifies the design of intelligent terminals.

The iSBX 270 module allows the user to add high level video display capability to his/her computer system with a minimal cost and effort. Typical applications for the iSBX 270 VDC include video displays for industrial operator stations, word processing systems, data base management products and many other uses.



October, 1981
Order Number: 210220-001

FUNCTIONAL DESCRIPTION

iSBX™ Interface

The iSBX 270 VDC interfaces to the Intel iSBX bus via the 8041A Universal Peripheral Interface (UPI) Microcomputer. The 8041A, under firmware control, provides communication between the base board and the iSBX 270 controller circuitry via the iSBX data and control lines. Data may be displayed immediately following power up, using default initialization provided by the 8041A UPI. In addition, eight high-level commands are provided by the iSBX 270 firmware; these eight commands are used to alter the default initialization of the controller and determine status. Following initialization, characters are displayed on the CRT by simply writing to the proper I/O port.

CRT Interface

The iSBX 270 VDC will interface to many B&W and RGB color display monitors. For B&W monitors, the iSBX 270 board provides TTL level signals for video, vertical sync and horizontal sync. Additionally, in B&W, two levels of intensity (normal and highlight) are supported under program control.

When operating in the color mode, the iSBX 270 module provides TTL level 75 ohm line drivers for Red, Green, and Blue Video and sync allowing 8 different colors to be displayed.

Composite video is not provided on the iSBX 270 MULTIMODULE board; however, with minimal external circuitry, composite video can be added (circuit design available; contact the local Intel Sales Office for details).

Table 1 lists several CRT vendors compatible with the iSBX 270 VDC.

Table 1. CRT's (B&W and Color)[1]

TYPE	VENDOR	MODEL #
B&W	Ball Brothers	TTL 120, TV 120, TV 50
	Motorola	M3570
	TSD	MDC-15
	ELSTON	DM30-12B0-51-A04
Color	Ball Brothers	7-015-0131
	IDT	19AC
	CONRAC	5711C13
	NEC	1202DH
	MITSUBISHI	C-3419

[1]**NOTE:** This in no way constitutes an endorsement by Intel Corporation of these companies' products. The companies listed are known to provide products compatible with the iSBX 270 board.

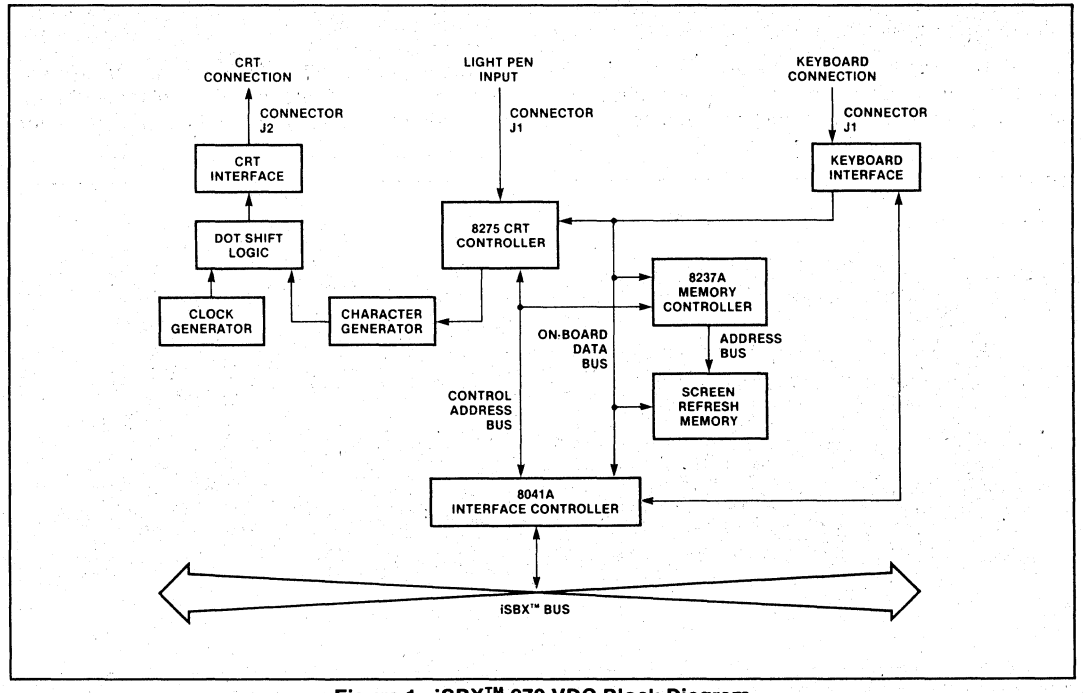


Figure 1. iSBX™ 270 VDC Block Diagram

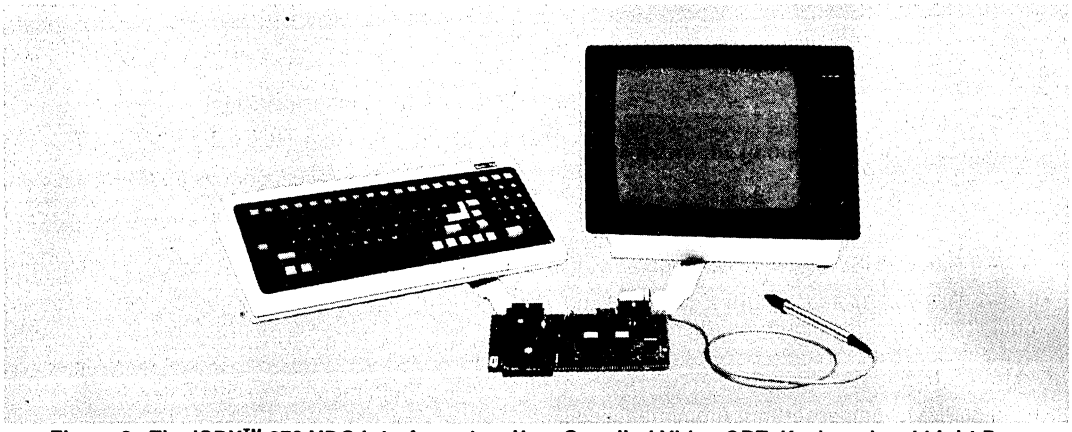


Figure 2. The iSBX™ 270 VDC Interfaces to a User-Supplied Video CRT, Keyboard and Light Pen

CRT Controller

The CRT Controller performs all timing and data buffering functions for the CRT. The iSBX 270 VDC uses the Intel 8275 CRT Controller (for additional details refer to the 8275 data sheet available from Intel).

Screen Refresh

The iSBX 270 VDC contains 4K bytes of high speed static RAM, as well as a high speed DMA controller (8237A). The 8237A, under the control of the 8041A UPI, takes care of both writing data to the screen and refreshing the screen.

Character Generation

The character fonts (128 characters, including alphabetic, numeric, and special characters) that are displayed on the CRT are stored in EPROM. The need may arise to display different character fonts, i.e., those used in international systems or custom symbols which are application specific. With the iSBX 270 VDC the user may modify any or all of the character fonts by simply reprogramming the EPROM. In addition, the user may utilize a larger EPROM to obtain up to 256 characters.

Keyboard Interface

The iSBX 270 VDC also interfaces to a keyboard I/O device via the J1 edge connector. The keyboard interface of the iSBX 270 VDC accepts up to eight TTL parallel data lines and one TTL strobe, either positive or negative. Keyboard input is indicated by a status bit in the 8041A and/or an interrupt. In addition, control lines are provided for visual and/or audible indicators.

Table 2 lists several keyboards that interface to the iSBX 270 VDC.

Table 2. Keyboards[1]

VENDOR	MODEL #
Advanced Input Devices	SK-067
Cherry	B70-05AB
Cherry	CB80-07AA
Chomerics	AN26109/AE26203
Cortron	35-500014
Keytronic	L1648
Keytronic	L1660
Keytronic	L1674-03
Keytronic	L1752
Microswitch	66SD6-7
Microswitch	87SD30-8

[1]**NOTE:** This in no way constitutes an endorsement by Intel Corporation of these companies' products. The companies listed are known to provide products compatible with the iSBX 270 board.

Light Pen Interface

Light pen I/O devices may be directly interfaced to the iSBX 270 VDC. A light pen hit is triggered on the rising edge of the light pen signal and is indicated by a status bit in the UPI 8041A and/or an interrupt.

Table 3 lists a light pen vendor whose product interfaces to the iSBX 270 VDC.

Table 3. Light Pens[1]

VENDOR	MODEL #
Information Control Co.	LP-700

[1]**NOTE:** This in no way constitutes an endorsement by Intel Corporation of this company's products. The company listed is known to provide products compatible with the iSBX 270 board.

SPECIFICATIONS

Controller Characteristics

DISPLAY

Programmable to a maximum of 35 rows x 80 columns of characters.

CRT OUTPUTS

B&W — TTL level HSYNC, VSYNC, Video.

Color — TTL level, 75 ohm line drivers for RGB and combined sync provide 8 different display colors.

FRAME RATE

50 Hz or 60 Hz via jumper settings (non-interlaced).

CHARACTER FONTS

5 x 7, 7 x 9 or 6 x 8 jumperable with appropriate crystal. Character generator uses 2716 EPROM. Also compatible with 2732A EPROM's. For generation of special fonts, please refer to iSBX 270 VDC Hardware Reference Manual.

VIDEO CONTROL

Reverse video, blinking, underline, highlight, cursor control and page or scroll mode.

TV MONITOR

Most video display monitors with a 10 MHz bandwidth or better.

LIGHT PEN INPUT

TTL level pulse, maximum 50 ns rise time, minimum 100 ns hold time.

Compatibility

CPU

Any iSBC single board computer or I/O board compatible with the MULTIBUS system bus and implementing the iSBX bus and connector.

Physical Characteristics

Width — 3.08 inches (7.82 cm)

Height — 0.8 inches (2.05 cm)

Length — 7.5 inches (19.05 cm)

Shipping Weight — 0.5 pounds (0.175 Kg)

Mounting — Occupies one double-wide iSBX MULTIMODULE position on boards; increases board height (host plus iSBX board) to 1.14 inches (2.90 cm).

Electrical Characteristics

Power Requirements +5 Vdc @ 1.3A.

Environmental Characteristics

Temperature — 0°C to 55°C (operating); −55°C to +85°C (non-operating).

Humidity — Up to 90% relative humidity without condensation (operating); all conditions without condensation or frost (non-operating).

Equipment Supplied

iSBX 270 VDC Controller
Reference Schematic

Cabling and connectors from the VDC controller to the CRT, keyboard and light pen are not supplied with the controller. Cables can be fabricated with commercially available cable and connectors as described in the iSBX 270 Hardware Reference Manual.

Reference Manual

143444-001 — iSBX 270 Video Display Controller Hardware Reference Manual (NOT SUPPLIED).

Reference manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, CA 95051.

ORDERING INFORMATION

Part Number	Description
SBX 270	Video Display Controller MULTIMODULE Board

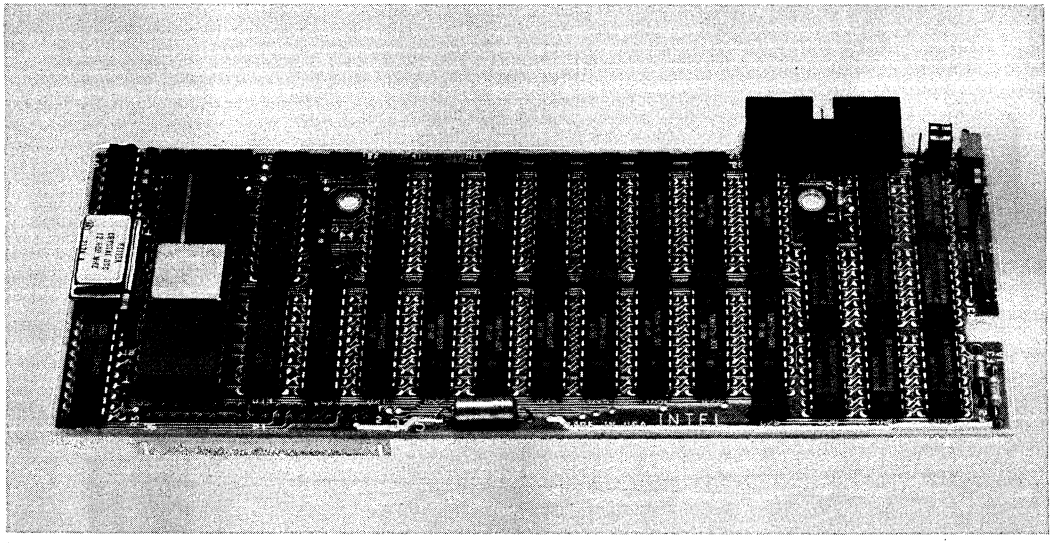
iSBX™ 275
VIDEO GRAPHICS CONTROLLER

- **Complete video graphics display controller on an iSBX™ MULTIMODULE™ board**
- **Interfaces to either black and white or color raster scan display monitors**
- **50 Hz or 60 Hz frame rate operation**
- **On-board refresh memory supports 512 x 512 black and white or 256 x 256 eight color display resolution**
- **High level drawing commands include line, arc, circle, rectangle, character, area fill, pan and scroll**
- **Includes Intel's 82720 Graphic Display Controller**
- **Compatible with industry standard iSBX™ bus interface**
- **Light pen interface**

The iSBX 275 Video Graphics Controller (VGC) allows the user to add high level video display capability to his/her computer system with minimal cost and effort. The iSBX 275 module provides a completely self-contained bit-mapped graphics subsystem on a 3″ x 7″ iSBX MULTIMODULE board. This same subsystem supports either black and white or eight color displays.

In addition, iSBX 275 VGC off-loads the system CPU from many of the graphics drawing functions. Under the control of the Intel 82720 Graphics Display Controller (GDC), the iSBX 275 board directly supports high level drawing commands which includes lines, arcs, circles, rectangles, characters, area fill, pan and scroll.

The iSBX 275 MULTIMODULE board is compatible with any computer board or system product supporting the industry standard iSBX bus; this includes most board and system products from Intel. Applications for the iSBX 275 VGC include video displays for industrial operator stations, engineering work stations, videotex, business presentation systems and other information display systems.



May, 1982
Order Number: 210506-001

FUNCTIONAL DESCRIPTION

iSBX™ Interface

The iSBX 275 VGC communicates with the host board through the iSBX bus. The iSBX bus is a standard I/O expansion bus interface (mechanical and electrical) for any microprocessor system. The iSBX standard interface allows system designers to optionally add incremental I/O functionality after the host microprocessor architecture is complete. In the case of the iSBX 275 VGC, the host board passes commands, data and status to and from the 82720 controller via two iSBX bus I/O ports.

The software interface consists of a series of high level commands passed to the 82720 controller. Table 1 contains a summary of 82720 software commands.

CRT Controller

The Intel 82720 is an intelligent graphics controller designed to be the heart of a raster-scan computer graphics display system. The 82720 performs all the basic timing needed to generate the raster display and manage the display memory. In addition, the 82720 supports several high level graphics figure drawing functions.

Table 2 lists several CRT vendors compatible with the iSBX 275 VGC.

Display Screen

The iSBX 275 VGC contains 32K bytes of high speed display memory, all of which is under the control of the 82720. The 82720 takes care of both writing and reading data to and from the screen and refreshing the screen.

The on-board display memory is organized as 16K words of 16-bits each. The 82720 reads or writes 16-bits of display data at a time. When displaying, the 82720 starts at the top left hand corner of the screen and sequences down the screen toward the bottom right hand corner.

In B&W mode all 16K, 16-bit words are treated as a contiguous block of memory, where a logical "1" in memory is displayed as an illuminated pixel.

In the color mode, three color planes, Red, Blue and Green, exist sequentially in memory but are displayed simultaneously. Each plane consists of 4K, 16-bit words where a logical "1" in a plane illuminates the corresponding color in that particular pixel.

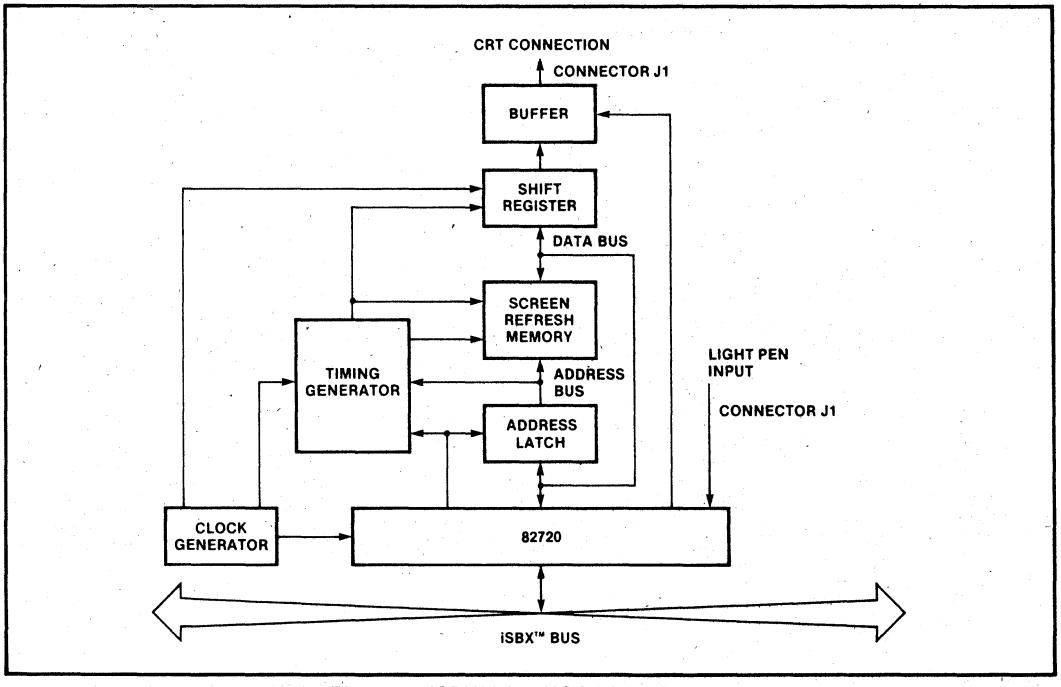


Figure 1. iSBX™ 275 VGC Block Diagram

Table 1. 82720 Command Summary

Video Control Commands	
RESET:	Resets the GDC to its idle state.
SYNC:	Specifies the video display format.
CCHAR:	Specifies the cursor and character row heights.
Display Control Commands	
START:	Ends idle mode and unblanks the display.
BCTRL:	Controls the blanking and unblanking of the display.
ZOOM:	Specifies zoom factors for graphics character writing.
CURS:	Sets the position of the cursor in display memory.
PRAM:	Defines starting addresses and lengths of the display areas and specifies the eight bytes for the graphics character.
PITCH:	Specifies the width of the X dimension of display memory.
Drawing Control Commands	
WDAT:	Writes data words or bytes into display memory.
MASK:	Sets the mask register contents.
FIGS:	Specifies the parameters for the drawing processor.
FIGD:	Draws the figure as specified above.
GCHRD:	Draws the graphics character into display memory.
Data Read Commands	
RDAT:	Reads data words or bytes from display memory.
CURD:	Reads the cursor position.
LPRD:	Reads the light pen address.

Table 2. CRT's (B&W and Color)[1]

Type	Vendor	Model #
B&W	Ball Brothers	TTL 120
	Motorola	M3570
	TSD	MDC-15
Color	Ball Brothers	7-015-0131
	IDT	19AC
	CONRAC	5711C13
	HITACHI	HM-2719/2713, HM-1719/1713
	NEC	1202DH
	MITSUBISHI	C-3419

[1]**NOTE:** This in no way constitutes an endorsement by Intel Corporation of these companies' products.

CRT Interface

The iSBX 275 VGC will interface to many B&W and RGB (Red, Green and Blue) color display monitors. For B&W monitors, the iSBX 275 board provides TTL level signals for video, vertical sync and horizontal sync or combined sync. When operating in the color mode, the iSBX 275 module provides TTL level 75 ohm line drivers for Red, Green, and Blue Video and a combined sync allowing 8 different colors to be displayed.

Composite video is not provided on the iSBX 275 MULTIMODULE board; however, with minimal external circuitry, composite video can be added (sample composite video circuit designs are included in the iSBX 275 Hardware Reference Manual).

Light Pen Interface

Light pen I/O devices may be directly interfaced to the iSBX 275 VGC. A light pen input or "hit" is triggered on the rising edge of the light pen signal and is indicated by a status bit in the 82720. The memory address of the light pen hit is obtained with a LPRD (Light Pen Read) command.

Table 3 lists a light pen vendor whose product interfaces to the iSBX 275 VGC.

Table 3. Light Pens[1]

Vendor	Model #
Information Control Co.	LP-700

[1]**NOTE:** This in no way constitutes an endorsement by Intel Corporation of this company's products.

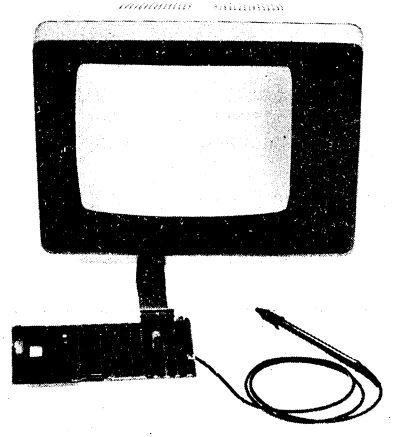


Figure 2. The iSBX™ 275 VGC Interfaces to a User-Supplied Video CRT and Light Pen

SPECIFICATIONS

Controller Characteristics

DISPLAY RESOLUTION

Black and White — nominal 512 x 512 x 1, interlaced

Color — nominal 256 x 256 x 3, non-interlaced

CRT OUTPUTS

Black and White — TTL level Video, HSYNC, VSYNC or CSYNC; maximum dot rate 13 MHz

Color — TTL level, 75 ohm line drivers for RGB and combined sync provide 8 different display colors with a 9.75 MHz maximum dot rate

FRAME RATE

50 Hz or 60 Hz via programmable option (non-interlaced)

VIDEO CONTROL

Pan and user selectable display and background color

DRAWING CONTROL

Lines, arcs, circles, rectangles, characters and area fill

CHARACTERS

Any user defined 8 x 8 font

MONITOR

Black and White — Most video display monitors with a TTL interface and a minimum bandwidth of 12 MHz

Color — Most video display monitors with a TTL interface and a minimum bandwidth of 6 MHz

LIGHT PEN INPUT

TTL level pulse, maximum 50 ns rise time, minimum 1.4 μS hold time

Compatibility

CPU

Any iSBC single board computer or I/O board compatible with the MULTIBUS system bus and implementing the iSBX bus and connector

Physical Characteristics

Width — 3.08 inches (7.82 cm)

Height — 0.8 inches (2.05 cm)

Length — 7.5 inches (19.05 cm)

Shipping Weight — 0.5 pounds (0.175 Kg)

Mounting — Occupies one double-wide iSBX MULTIMODULE position on boards; increases board height (host plus iSBX board) to 1.14 inches (2.90 cm)

Electrical Characteristics

Power Requirements — +5 Vdc @ 1.5A

Environmental Characteristics

Temperature — 0° to 55°C (operating); −55°C to +85°C (non-operating)

Humidity — Up to 90% relative humidity without condensation (operating); all conditions without condensation or frost (non-operating)

Equipment Supplied

iSBX 275 VGC Controller

Reference Schematic — Cabling and connectors from the VGC controller to the CRT and light pen are not supplied with the controller. Cables can be fabricated with commercially available cable and connectors as described in the iSBX 275 Hardware Reference Manual.

Reference Manual

144829-001 — iSBX 275 Video Graphics Display Controller Hardware Reference Manual (NOT SUPPLIED)

Reference manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, CA 95051.

ORDERING INFORMATION

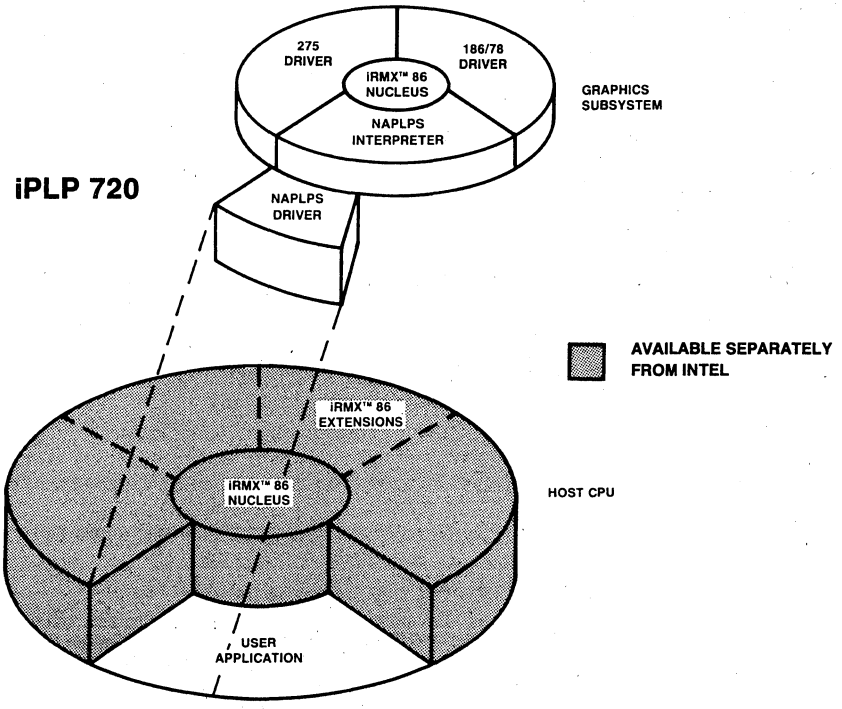
Part Number	Description
SBX 275	Video Graphics Display Controller MULTIMODULE Board

iPLP 720
NAPLPS Interpreter

- **Provides decoding of NAPLPS (Videotex) commands**
- **Multiple font sizes and definitions**
- **Full iRMX™ 86 compatibility**
- **Compact for EPROM installation**
- **Compatible with ANSI BSR X3.110 -1983**
- **Driver support for Intel Graphics hardware modules**
- **Complete library of high-level frame management instructions**
- **Resolution independant presentation**
- **Simplified communication of Graphic images between systems**

The Intel iPLP 720 NAPLPS (North American Presentation Level Protocol Syntax) Interpreter provides both a powerful library of high-level commands, and the drivers necessary to support the iSBX™ 275 or iSBC 186/78 Graphics Modules in an iRMX™ 86 Operating System (Release 5 or later) environment. It allows the OEM to quickly tailor an Intel system for application into the rapidly growing Videotex marketplace. Individual iSBC® modules may also be configured from Intel's broad product family, and iPLP 720 will also be convenient for the those implementing custom, component designs based on the Intel 82720 Graphics Display Controller (GDC).

Regardless of the hardware configuration, iPLP 720 is the most powerful and efficient product available that brings full ANSI X3L2 Videotex compatibility to an iRMX 86 environment.



May, 1984
Order Number: 280001-001

FUNCTIONAL DESCRIPTION

NAPLPS Interpreter

The iPLP 720 software implements the NAPLPS videotex standard on any 82720-based graphics application using the iRMX 86 Operating System, Release 5 or later. The NAPLPS Standard is a significant advancement in device-level graphics software. It creates a predictable environment for development of presentation frames along with a transmission scheme to dramatically improve performance when moving graphics images from one system to another. The software supports two environments: stand-alone and distributed, depending on the hardware configuration.

All elemements of iPLP 720 can run as tasks of the operating system or as part of the graphics application program, hence a stand-alone partitioning of graphics activities such as with the iSBX 275 MULTIMODULE™. In a distributed environment, the device driver runs under the iRMX 86 Operating System and the remaining controller code and NAPLPS interpreter are exercised by a separate processor dedicated to graphics activities. The iSBC 186/78 subsystem was designed especially for the distributed solution.

This architecture also allows iPLP 720 to run under non-iRMX environments.

iSBC® 186/78 Video Graphics Subsystem Support

By virtue of its on-board, high integration microprocessor (the Intel 80186), the iSBC 186/78 subsystem is an excellent platform on which to perform graphics routines in a distributed environment. This is particularly important in multi-user systems where one iSBC 186/78 subsystem can be dedicated for each user. The host CPU board is thereby off-loaded to direct more global system level operations such as database management or network communications.

The combination of the iSBC 186/78 subsystem and iPLP 720 interpreter meets all requirements of ANSI's NAPLPS Standard Reference Model (SRM).

iSBX™ 275 Graphics Controller Support

In single-user applications or where graphics activities are not the major focus of the system, the iSBX 275 MULTIMODULE shares the CPU on the host processor board through the iSBX Expansion Bus. The subsystem formed in this manner supports either monochrome or eight colors and is a very cost-

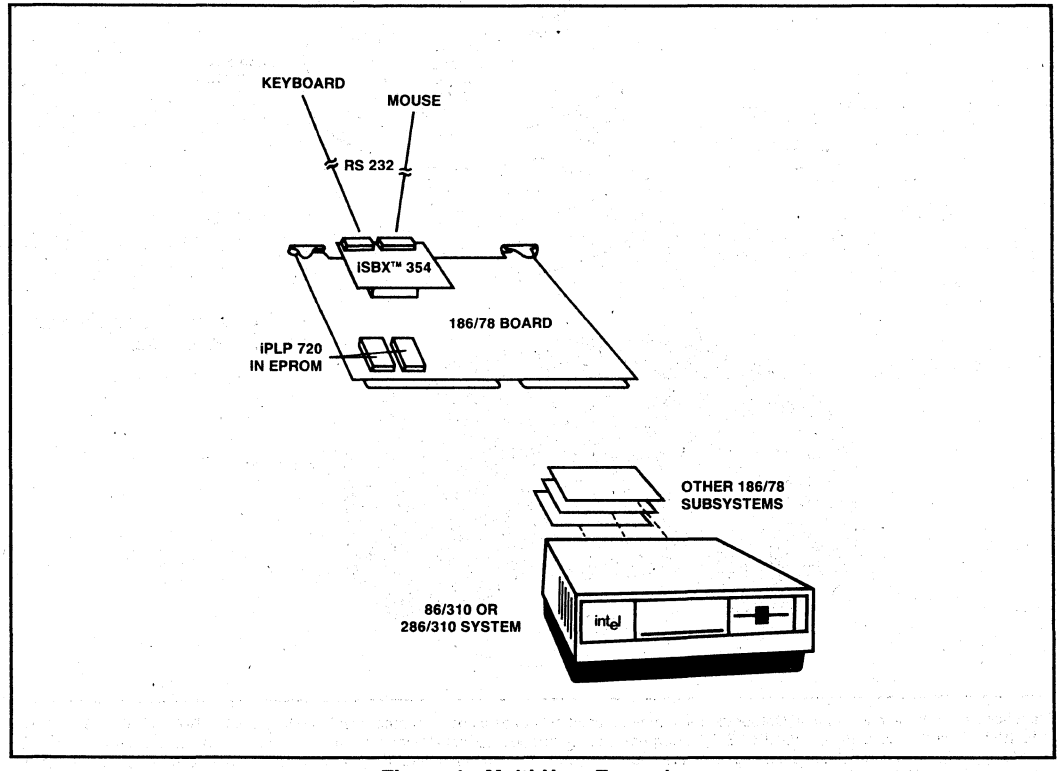


Figure 1. Multi-User Example

Order Number: 280001-001

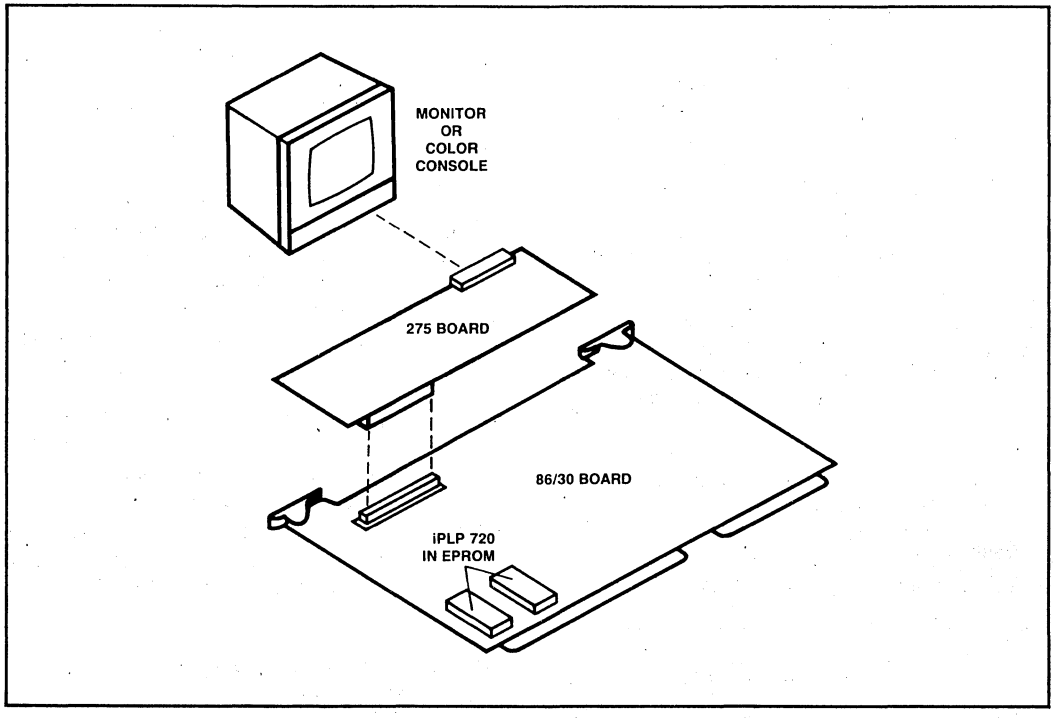


Figure 2. Single-User Example

effective solution. Like the iSBC 186/78 subsystem, this expansion module is based on the Intel 82720 component.

82720 Component Designs

The Intel 82720 GDC is an intelligent graphics controller component designed to operate as the heart of a raster-scan computer graphics display system. The 82720 performs all the basic timing needed to generate the raster display and manage the display memory. In addition, it supports several high-level graphics figure drawing functions. The Intel 82720 is an alternative to the NEC 7220 component. Custom hardware designs based on either component are supported by iPLP 720 running in an iRMX 86 environment.

NAPLPS COMMAND LIBRARY

In addition to providing driver support for Intel's growing family of graphics products, the iPLP 720 NAPLPS Interpreter decodes a wealth of high level commands to streamline the development of application code for videotex environments.

The NAPLPS standard provides character set encodings of high level text and graphics commands and capabilities. The iPLP 720 NAPLPS Interpreter implements every one of these character encodings.

The following table lists the major high level commands which are encoded in the NAPLPS standard:

Table 1. iPLP 720 Command Library

Geometric Drawing Primitives:	
Point	Incremental Point
Line	Incremental Line
Polygon	Incremental Polygon
Arc	Rectangle
Text:	
Character Rotation	Cursor Styles
Character Path Movement	Word or Character Wrap Around
Inter-character Spacing	Inter-row Spacing
Character Field Dimensions	Scrolling
Texture:	
Line Texture	
Highlighting	
Programmable Texture Patterns	
Texture Mask Size	
Miscellaneous Functions:	
Logical Pel Size	Set Color
Macro Definitions	Mosaic Sets
Blink Processes	Wait Intervals
Dynamically Redefinable Character Sets	

Order Number: 280001-001

Other features are configurable as defined in the standard, however they are typically device dependent and therefore reflect the users application. Consequently, the standards document should be consulted to assure compatibility.

ANSI X3L2 NAPLPS Specification

The development of any standard is an evolutionary process. Consequently, the specification used as a reference model for the design of iPLP 720 was published in October 1982. Updates to iPLP 720 will render it fully compatible with the final specification. The American National Standards Institute (ANSI) administers the standard specification and makes it available to all interested parties. Requests for copies should be directed to:

X3 Secretariat
Computer Business Equipment Manufacturers Association (CBEMA)
311 First Street, NW
Washington, D.C. 20001

Reference document: BSR X3.110

iPLP 720 Specifications

Code size — 90 Kbytes (four 27256 EPROMs) in distributed mode;
— 72 Kbytes in stand-alone mode

Source-code language — PL/M 86

Related Literature

Reference material may be ordered from any Intel sales representative, distributor office, or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, Calif., 95051.

146144 — iPLP 720 Software Reference Manual

210506 — iSBX™ 275 Video Graphics Controller Data Sheet

231035 — iSBC® 186/78 Video Graphics Subsystem Data Sheet

146666-001 — 186/78 Reference Manual.

210655 — 82720 GDC Component Data Sheet

9803126 — iRMX™ 86 Configuration Guide

145412 — Intel's Guide to Understanding the ANSI Videotex/Teletex Standard

ORDERING INFORMATION

Intel makes available a variety of licenses to the iPLP 720 NAPLPS interpreter that allow different plans for incorporation of the Intel software into the final product. The Intel Master Software Agreement should be consulted to determine which is best suited for the particular application and production environment.

The iPLP 720 NAPLPS Interpreter comes in two formats as shown below, along with yearly update services. The iRMX 86 Real-time Multitasking Operating Systems are available separately.

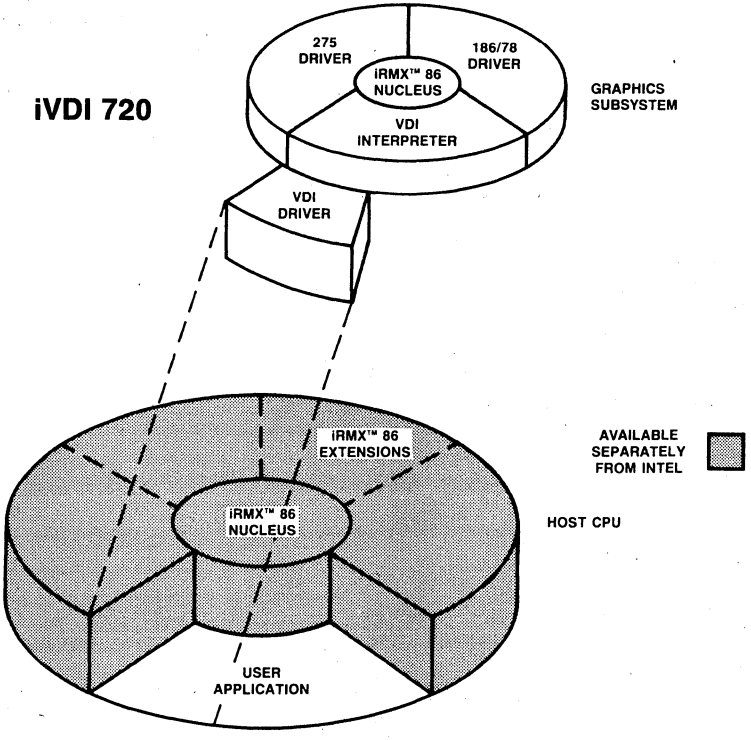
iPLP 720RO	OEM License, single density (both ISIS and iRMX formats are included)
iPLP 720RF	Incorporation fee payment
iPLP 720WX	Object code update

iVDI 720
VIRTUAL DEVICE
Interpreter

- **Provides standardized decoding of high-level graphics commands**
- **Full iRMX™ 86 compatibility operating system (Rel. 6)**
- **Standardized input & output drivers**
- **Compact for EPROM installation**
- **Support for iSBX™ 275 and iSBC® 186/78 Graphics hardware modules**
- **Procedural interface from Pascal 86, PL/M 86 and Fortran 86**
- **Compatible with (proposed) ANSI X3H33 specification**
- **Virtual Device Metafile interpreter**

The Intel iVDI 720 Graphics Virtual Device Interpreter provides both a powerful library of high-level commands, and the drivers necessary to support the iSBX™ 275 or iSBC® 186/78 graphics modules in an iRMX™ 86 (Release 6) environment. It allows the OEM to quickly tailor an Intel system for application into the rapidly growing graphics marketplace, especially low-cost CAD/CAE, CAM, and process control. Individual single-board computer (SBC) modules may also be configured from Intel's broad product family.

For intra-systems graphics control, iVDI 720 is the most powerful and efficient product available that brings (proposed) ANSI X3H33 compatability to an iRMX 86 operating system environment.



JUNE, 1984
ORDER NUMBER: 280002-001

FUNCTIONAL DESCRIPTION

Graphics Standard Software

The iVDI 720 Graphics Virtual Device Interpreter implements the proposed ANSI standard on any Intel-based graphics system running under the iRMX 86 operating system, release 6. The proposed standard is a significant advancement in graphics software. It creates a predictable environment for the input and output of high-level commands between the user and system, or among the graphics peripherals attached to the system, such as a mouse, tablet, printer or plotter. The software supports two environments: stand-alone and distributed, depending on the hardware configuration.

All elements of iVDI 720 can run as tasks of the operating system or as part of the graphics application program, hence a stand-alone partitioning of graphics activities such as with the iSBX 275 MULTIMODULE™ attached to a general purpose CPU board like the iSBC 86/30. In a distributed environment, the device driver runs under the iRMX 86 operating system and the remaining application code and VDI interpreter are exercised by a separate processor dedicated to graphics activities. The iSBC 186/78 subsystem was designed especially for the distributed solution.

iSBC® 186/78 Graphics Subsystem Support

By virtue of its on-board, high integration microprocessor (the Intel 80186), the iSBC 186/78 subsystem is an excellent platform on which to perform graphics routines in a distributed environment. This is particularly important in multi-user systems where one iSBC 186/78 subsystem can be dedicated to each user. (see figure 1)

The compact coding of the iVDI 720 Graphics Virtual Device Interpreter lends itself to EPROM installation on the iSBC 186/78 subsystem. The host CPU board is thereby off-loaded from graphics activities so it can direct more global system level operations such as database management or network communications.

iSBX™ 275 Graphics MULTIMODULE™ Support

In single-user applications or where graphics activities are not the major focus of the system, the iSBX 275 MULTIMODULE shares the CPU on the host processor board through the iSBX expansion bus. The subsystem formed in this manner supports either monochrome or eight colors and is a very cost-effective solution. Like the iSBC 186/78 subsystem,

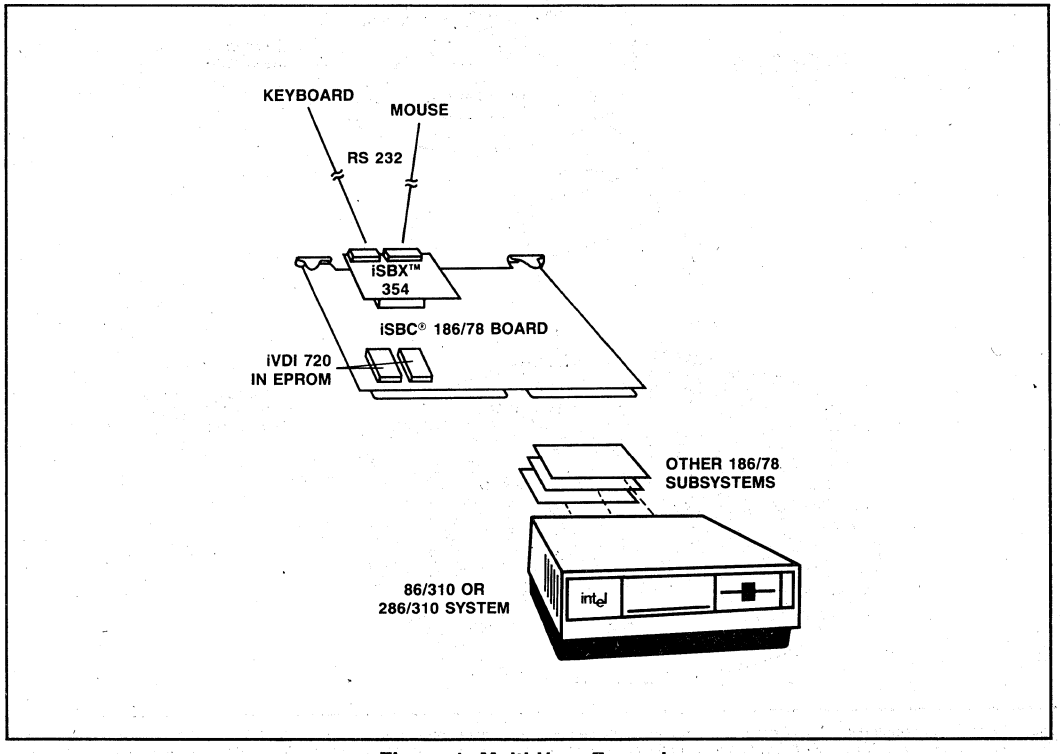


Figure 1. Multi-User Example

this expansion module is based on the Intel 82720 Graphics Display Controller (GDC) component. (see figure 2)

For example using an iSBC 86/30 CPU board, the iVDI 720 library can be installed in EPROM to simplify the application and provide higher performance execution.

82720 Component Designs

The Intel 82720 GDC is an intelligent graphics controller component designed to operate as the heart of a raster-scan computer graphics display system. The 82720 performs all the basic timing needed to generate the raster display and manage the display memory. In addition, it supports several high-level graphics figure drawing functions. The Intel 82720 is an alternative to the NEC 7220 component.

VDI COMMAND LIBRARY

In addition to providing driver support for Intel's growing family of graphics modules, the iVDI 720 Graphics Virtual Device Interpreter decodes a wealth of high-level commands to streamline the development of application code for a variety of graphics devices.

The proposed ANSI standard provides multiple encodings of high-level text and graphics commands and capabilities. The iVDI 720 software decodes a binary representation of these proposed commands, along with the Virtual Device Metafile (VDM) routines that allow consistant formatting and storage of VDI encoded images.

In addition to a full set of inquiry functions, many additional high-level commands are supported in the iVDI 720 software. (See Table 1)

These features are configurable as defined in the iVDI 720 Software Reference Manual. However, they are typically device dependent and therefore reflect the users application. Consequently, the reference manual should be consulted to assure compatability.

DEVELOPMENT ENVIRONMENT

Intel's family of development systems and their extensions are highly recommended for both the development of iVDI 720 and related application code. Languages that are supported include Fortran 86, Pascal 86 and PL/M 86. All iVDI 720 commands can be called from any of these programming languages through the PL/M 86 procedural interface that is integral to the iVDI 720 product.

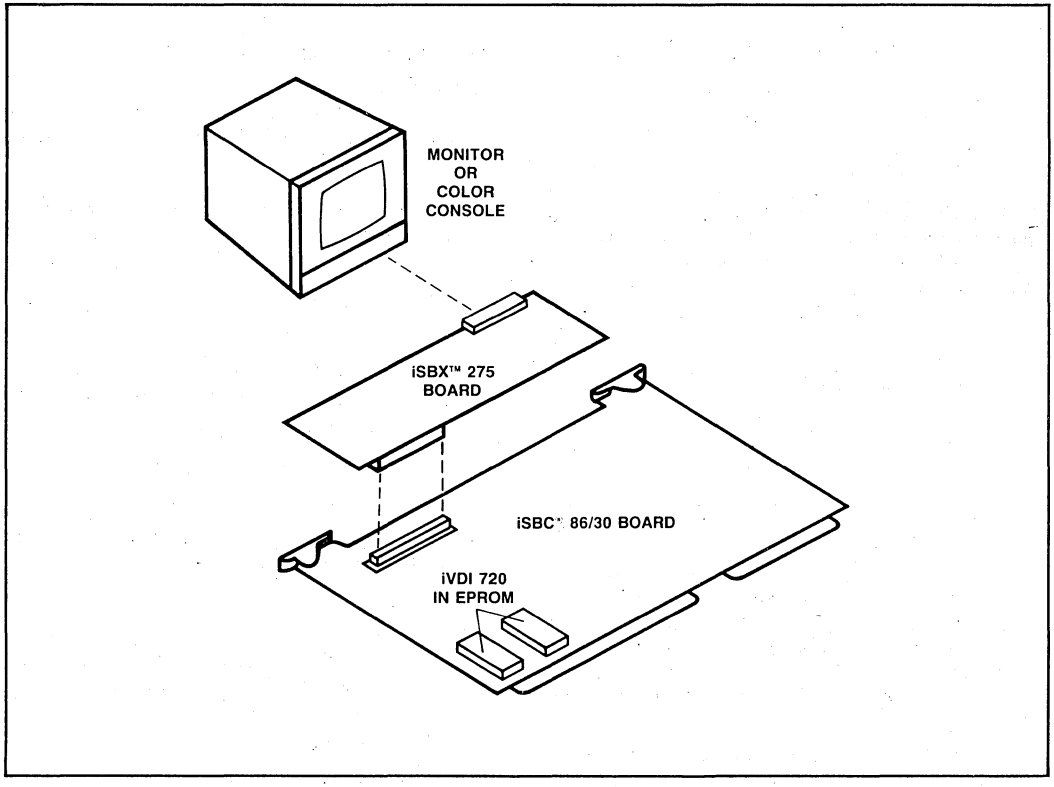


Figure 2. Single-User Example

Table 1. iVDI 720 Command Library

Graphical Elements:
- Polyline
- Polymarker
- Polygon
- Circle
- Arc
- Arc Close (Pie or Chord)
- Text
- Append Text
- Cell Array

Attribute Elements:
- Aspect Source Flags
- Bundled & Individual Attributes
- Character Orientation
- Character Height
- Character Path
- Character Expansion Factor
- Character Spacing
- Text Alignment
- Perimeter Type & Color
- Interior Style
- Hatch Fill
- Marker Type & Color
- Pattern Fill
- Line Type & Color
- Pattern Definition
- Set Color Table
- Text Precision
  - String
  - Character
  - Stroke
- Pattern Size
- Pattern Reference Point
- Text Color

Control & Descriptor Elements:
- Begin & End Metafile
- Begin & End Picture
- Background Color
- VDC Extent
- Clip Rectange
- Clip Indicator
- Clear Surface
- Defaults Replacement
- Set Device Viewpoint
- Color Direct Precision
- Scaling Mode
- Color Specification Mode
- Marker Size Mode

Input Elements:
- Initialize Locator
- Initialize String
- Sample Locator
- Sample String
- Request Locator
- Request Locator
- Set Prompt State
- Set Echo State
- Release Input Device
- Set Input Device Mode

SPECIFICATIONS

ANSI X3H33 VDI Specification

The American National Standards Institute (ANSI) administers the standard specification. Requests for information should be directed to:

X3 Secretariat
Computer Business Equipment Manufacturers Association (CBEMA)
311 First Street, NW
Washington, D.C. 20001

Intel is heavily involved in the development of the ANSI X3H33 Virtual Device Interface standard. We will endeavor to bring to our user base the latest revisions through phased introductions and updates. Consequently, it is strongly advised that implementers of iVDI 720 also subscribe to the update service (VDI 720 WX, see below).

iVDI 720 Specifications

Code size — 80 Kbytes in distributed mode (using the iSBC 186/78 subsystem), including the iRMX 86 nucleus

Code size — 64 Kbytes in stand-alone mode (using the iSBX 275 MULTIMODULE)

Source-code language — PL/M 86

Related Literature

Reference material may be ordered from any Intel sales represenative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, Calif., 95051.

146717 — iVDI 720 Software Reference Manual

210506 — iSBX 275 Video Graphics Controller Data Sheet

231035 — iSBC 186/78 Video Graphics Subsystem Data Sheet

146666 — iSBC 186/78 Video Graphics Subsystem Hardware Reference Manual

210655 — 82720 GDC Component Data Sheet

9803126 — iRMX 86 Configuration Guide

Ordering Information

Intel makes available a variety of licensing programs to the iVDI 720 Graphics Virtual Device Interpreter which allow different plans for incorporation of the Intel software into the final product. The Intel Master software Agreement should be consulted to determine which plan is best suited for the particular application and production environment.

The iVDI 720 Graphics Virtual Device Interpreter comes in three formats as shown below, along with source listings and update services. The iRMX 86-Real-time Multitasking Operating System is available separately.

iVDI 720RO OEM license (8 inch single-sided/double densite ISIS and iRMX plus 5¼ inch double-sided/double density iRMX formats are supplied)

iVDI 720RF Incorporation fee payment

iVDI 720WX Object code update

intel

APPLICATION NOTE

AP-180

May 1984

Graphics Applications of the iSBX™ 275 Graphics Controller

Brad Janeway

Order Number: 280008-001

GRAPHICS APPLICATIONS OF THE iSBX™ 275 GRAPHICS CONTROLLER

CONTENTS

GENERAL INTRODUCTION

The iSBX™ 275 video graphics controller is a double-wide iSBX MULTIMODULE™ board. (Refer to Fig. 1.) It provides low-cost graphics capability for boards and systems which support the iSBX bus interface. This graphics controller simplifies graphics drawing for the user while offloading the central processing unit (CPU) of the system. Applications for the iSBX 275 graphics controller include video displays for industrial operator stations, engineering work stations, videotex, business presentation systems, and other information display systems.

The purpose of this application note is to show how easily graphics drawing can be implemented using the iSBX 275 graphics controller. For this discussion, familiarity with Intel products is assumed, including iRMX™ 86 file structure, PL/M-86 code generation, and Intel boards and board jumpering procedure. A general knowledge of algebra and geometry is also assumed, but prior graphics experience is not required.

A dual function onboard processor controls the iSBX 275 board. (Refer to Fig. 1.) This processor, the 82720 Graphics Display Controller (GDC), is both a graphics drawing processor and a display processor. The iSBX 275 board provides 16K words (32K bytes) of onboard static RAM display memory for use by the GDC. All calculations and addressing for drawing the requested figure in display memory are performed by the GDC. The GDC also maps this display memory onto the monitor screen and performs screen refresh. Refer to the 82720 GRAPHICS DISPLAY CONTROLLER Data Sheet, Intel order number 210655-002, for information on the GDC component.

The iSBX 275 board provides the capability for drawing points, lines, arcs, circles, and rectangles in a user-defined line style. It also provides for drawing characters and special user-defined symbols. In addition, rectangular areas can be filled with a user-specified pattern.

Jumpers on the iSBX 275 board enable operation in either black-and-white or eight-color display mode. The iSBX 275 board supports a black-and-white display resolution of 512 × 512 or a color display resolution of 256 × 256. For jumper configuration information, refer to the iSBX 275 VIDEO GRAPHICS CONTROLLER MULTIMODULE BOARD REFERENCE MANUAL, Intel order number 144829-001, pages 2-4 through 2-7.

GENERAL OVERVIEW

This application note begins with some background information related to raster-scan graphics. It then describes a set of PL/M-86 procedures which provide a simple interface to the iSBX 275 board. Next, two graphics applications are discussed which utilize these graphics procedures. The first application uses the black-and-white display mode of the iSBX 275 board, while the second utilizes the color display mode of the board. Finally, some performance considerations are addressed.

Listings of the applications described in this application note are contained in an appendix. The complete source code for the graphics procedures and the airplane application are available from INSITE.

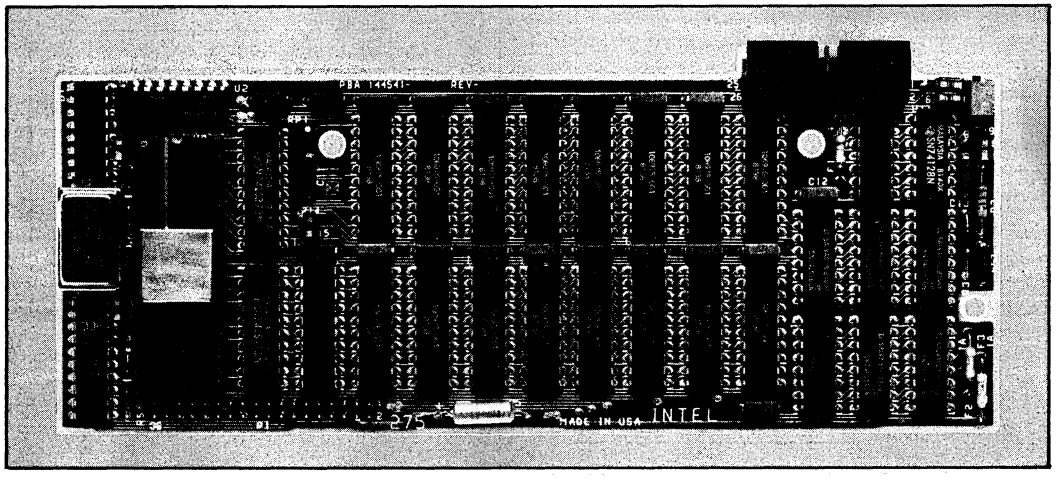


Figure 1. iSBX 275 Graphics Controller

BACKGROUND INFORMATION

Overview

This section first provides information on bit-mapped graphics. Then raster-scan monitors are discussed.

Bit-Mapped Graphics

A monitor screen can be thought of as a rectangular grid of picture elements, or pixels. This grid is defined by an aspect ratio x:y, where x is the horizontal dimension and y is the vertical dimension, in pixels. The typical monitor aspect ratio is 4:3.

An RGB (Red-Green-Blue) color monitor has three electron guns, one for each of the primary colors red, green, and blue. These guns are used for drawing on the monitor screen. A black-and-white monitor has one gun.

The iSBX 275 board contains 16K words or 256K bits of bit-mapped display memory. The GDC addresses this memory, often referred to as the bit map, as 16K words of memory. (See Fig. 2.) Each bit (or combination of 3 bits in color display mode) is mapped by the GDC to a unique pixel on a monitor screen. In black-and-white display mode, all 256K bits lie in one plane, so each bit represents one pixel. In color display mode, the bit map is divided into four color planes, with 4K words (64K bits) in each plane. These planes logically overlap each other, as shown in Fig. 3. One of these planes is used for each of the primary colors red, blue, and green. The bits from the red color plane are input to the red electron gun of an RGB monitor. The bits from the other two color planes are likewise input to the respective guns of the monitor. The color of a pixel on a monitor screen is thus determined by three bits, one from each plane. This results in a palette of eight drawing colors, ranging from all three bits of a pixel OFF (black) to all three bits ON (white).

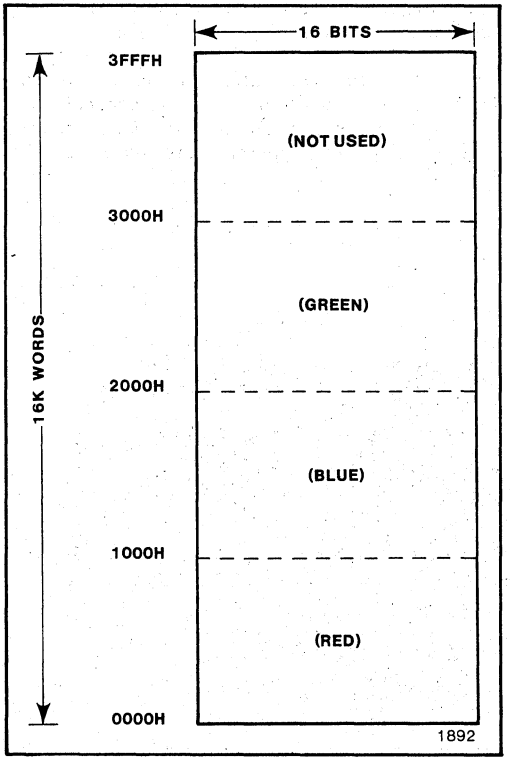


Figure 2. GDC Addressing

In order for the GDC to properly map bits in display memory onto the monitor screen, the bit map must be configured to match the desired display image. Configuring the bit map for a 1:1 aspect ratio provides the most efficient use of the display memory on the iSBX 275 board, as all of the bits in the display memory are utilized. As shown in Fig. 4a and Fig. 5a, this results in a black-and-white resolution of 512 × 512 or a color resolution of 256 × 256 pixels. However, an aspect ratio of 4:3 is typically desired, and any reasonable aspect ratio may be used. In any configuration, the horizontal dimension must be a multiple of 32 pixels. The black-and-white configuration shown in Fig. 4b leaves a minimum of 64 bits (4 words) in the bit map unused, and the color configuration shown in Fig. 5b leaves a minimum of 160 bits (10 words) unused. In these figures, the shaded region represents an exact 4:3 aspect ratio, resulting in

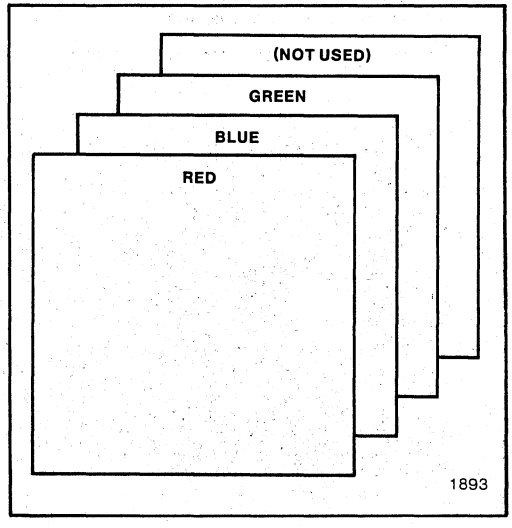


Figure 3. Color-Plane Orientation

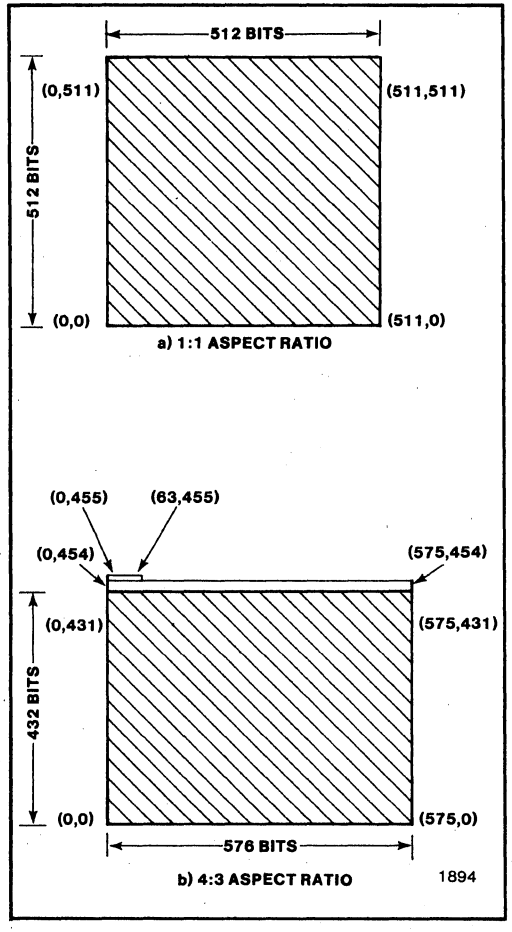


Figure 4. Black-and-white Mode Configuration

13312 bits (832 words) or 3328 bits (208 words) remaining unused for the black-and-white or color configuration, respectively.

A logical x,y address is associated with each drawing command sent by user code to the iSBX 275 board. Fig. 4 and Fig. 5 show the range of logical addresses for some typical bit map configurations. The logical coordinates associated with a drawing command are translated into the appropriate bit map word address (or addresses, in color mode). This is accomplished by the iSBX 275 routines described in the PL/M-86 GRAPHICS LIBRARY section. The iSBX 275 board then sets the appropriate display memory bits.

Raster-Scan Monitors

The iSBX 275 board utilitzes and RGB monitor for color displays and either an RGB or a black-and-white monitor for monochrome displays. An RGB monitor has three phosphors and three electron guns, one pair for each of the primary colors red, blue, and green, for generating images on its screen. For a color display using the iSBX 275 board, the bits from the red, blue, and green color planes in display memory are fed to the respective electron gun inputs of an RGB monitor. For a black-and-white display using and RGB monitor, the bits from display memory are typically fed to the green gun input of the monitor. A black-and-white monitor has one electron gun and one phosphor for generating images on its screen.

A black-and-white monitor typically has two synchronization inputs, one for horizontal sync and one for vertical sync. An RGB monitor typically has one sync input, called Combined SYNC (CSYNC), which is a combination of these two signals.

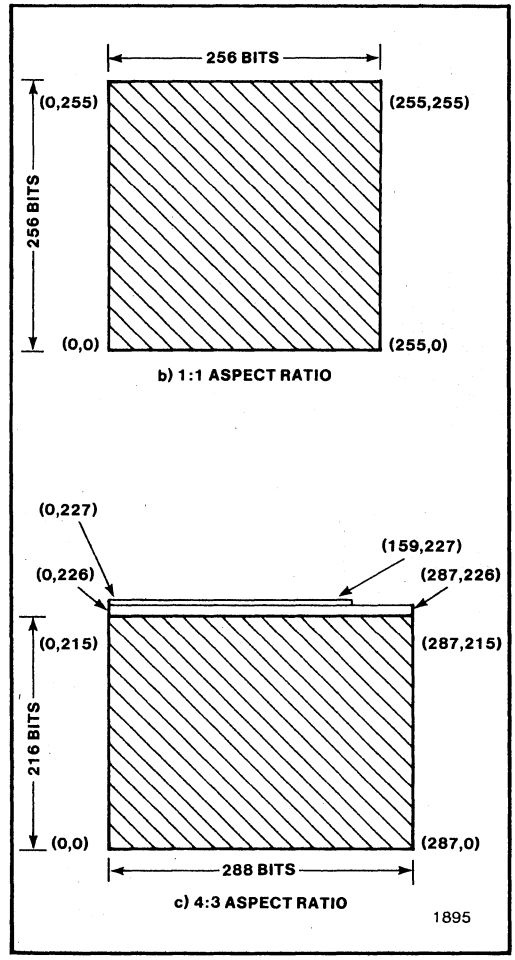


Figure 5. Color Mode Configuration

The following discussion deals with the manner in which a raster-scan monitor generates an image on its screen. Monitor timing is described, providing an understanding of the initialization parameters which are required by the iSBX 275 board. For simplicity, the discussion assumes a black-and-white monitor.

CRT

The inside surface of a cathode ray tube (CRT) is coated with a light-emitting phosphor. A raster-scan monitor displays an image by scanning an electron gun across the face of the CRT, moving from left to right and from top to bottom (as viewed from the front of the monitor). For each pixel which is to be turned ON, the gun generates an electron beam of sufficient energy to cause the pixel to glow brightly. However, the persistence of the phosphor is typically only twenty milliseconds, requiring that the phosphor be struck at regular intervals in order to keep it glowing brightly. This is accomplished by continuously scanning the entire face of the CRT as described below.

RASTER-SCAN

A complete display image on a monitor screen is called a frame. It consists of one or two fields of information, each generated by one raster-scan cycle. (Refer to the section on INTERLACING.) A raster-scan cycle consists of a vertical retrace-and-scan and a number of horizontal retrace-and-scans. Upon receiving a vertical syncronization pulse from the iSBX 275 board, a monitor begins a raster-scan cycle. The beginning of a frame is defined by the coincidence of a vertical and a horizontal sync pulse from the iSBX 275 board.

In response to a vertical sync pulse, a monitor first executes a vertical retrace, moving its electron gun rapidly to the top of the CRT. Then the monitor performs a vertical scan. It moves its electron gun back down the face of the CRT at a constant, slower rate, arriving at the bottom just as the next vertical sync pulse arrives. The monitor expects a vertical sync pulse at regular intervals, defined by the field rate of the monitor. The typical field rate is 60 Hz in the U.S. and 50 Hz in Europe. The length of the sync pulse must be within the limits specified by the monitor.

Upon receiving a horizontal sync pulse, a monitor first executes a horizontal retrace, moving its electron gun rapidly to the left of the CRT. The monitor then performs a horizontal scan. It moves its electron gun back across the face of the CRT at a constant, slower rate, arriving at the right side just as the next horizontal sync pulse arrives. The monitor expects a horizontal sync pulse at regular intervals, defined by the horizontal rate of the monitor. The typical horizontal rate is 15.75 KHz in the U.S. and 15.63 KHz in Europe. The length of the sync pulse must be within the limits specified by the monitor.

The path which the electron gun traces across the face of the CRT, due to the combination of vertical and horizontal retrace-and-scan cycles, is similar to that shown in Fig. 6 and Fig. 7. The dotted lines represent retrace and the solid lines represent horizontal scan. All of the horizontal scan lines traced during a vertical scan comprise a field. The number of lines in a field is a function of the field rate and the horizontal rate of the monitor. Fig. 6 shows the case for which a frame consists of one field, and Fig. 7 shows the case for which a frame consists of two interlaced fields.

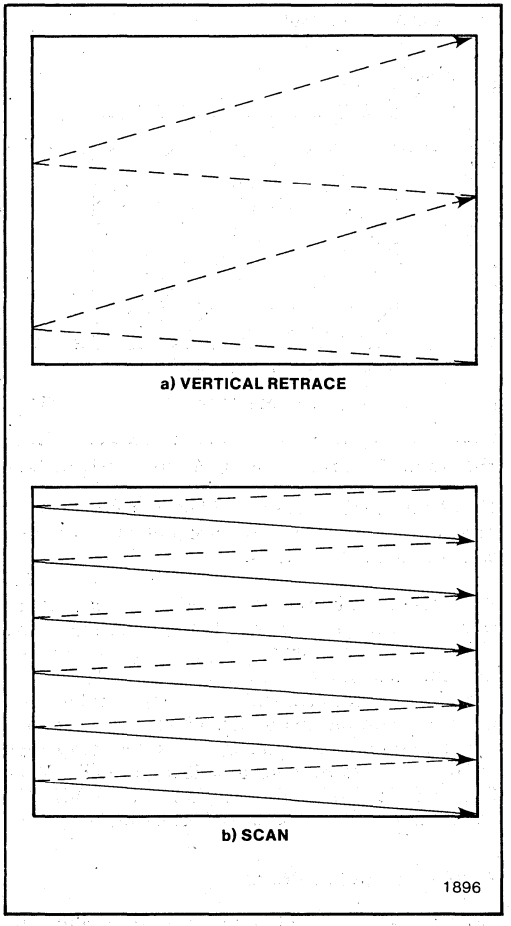


Figure 6. Non-interlaced Raster Scan Frame = Field

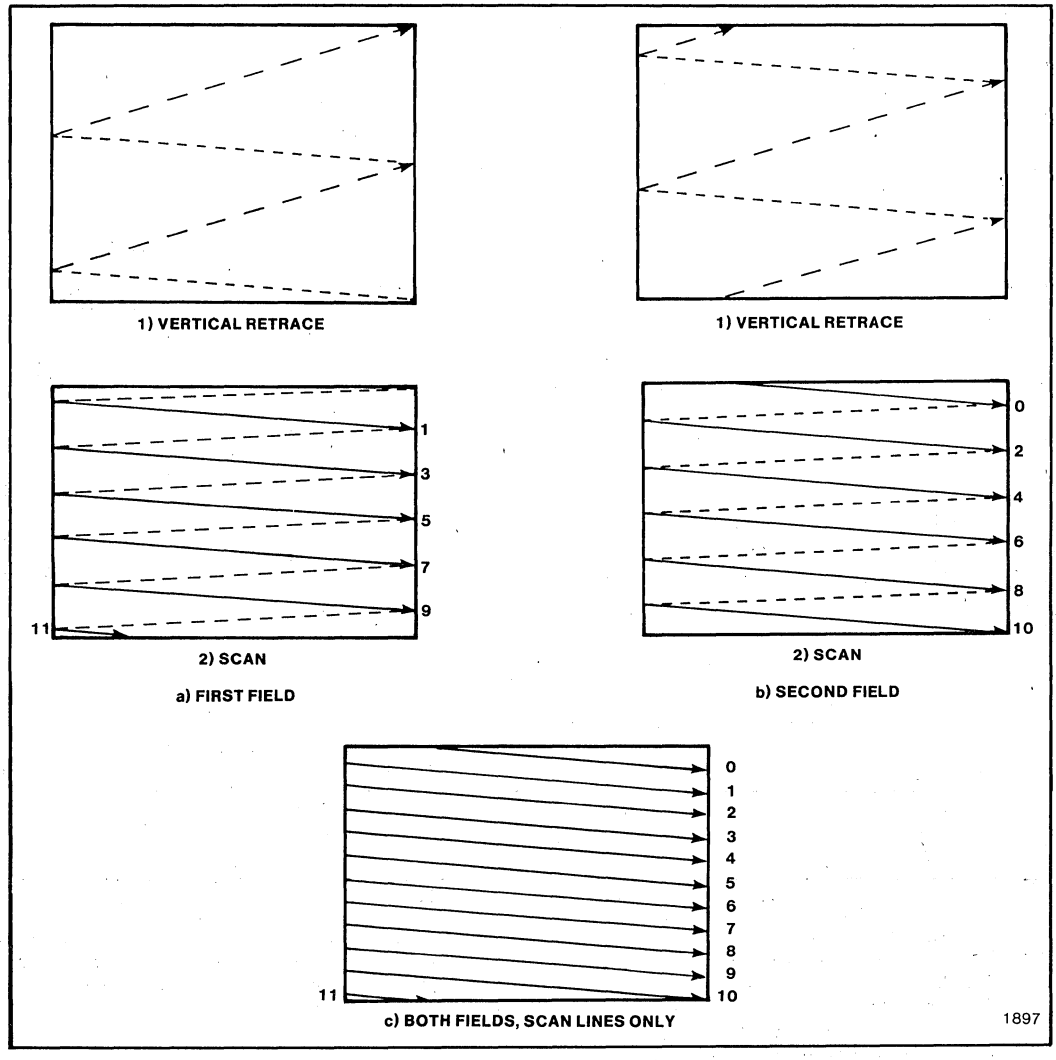


Figure 7. Interlaced Raster Scan Frame = 2 Fields

During the active diplay portion of each horizontal scan, the display memory bits mapped to that scan line are input to the electron gun of the monitor. The electron gun interprets each bit as beam intensity. A logical "1" bit causes the gun to excite the phosphor at that pixel location, such that it emits light, whereas a logical "0" bit leaves the pixel OFF. The rate at which these bits are input to the gun, and hence the rate at which pixels are drawn on the monitor screen, is referred to as the bandwidth of the board. For the iSBX 275 board, this bandwidth is determined by the frequency of the oscillator on the board. In the divide-by-two mode of the iSBX 275 board, this bandwidth is half of the oscillator frequency. The board bandwidth must be within the bandwidth range of the monitor.

In order to avoid leaving unwanted traces across the display, the electron gun must be off, or blanked, during retrace. Since a discrete time period is required to turn off an electron beam, a monitor specifies minimum horizontal and vertical front porch blanking times. (Refer to Fig. 8.) During a front porch blanking time, which immediately precedes the corresponding sync pulse, the monitor turns off its electron gun. The front porch

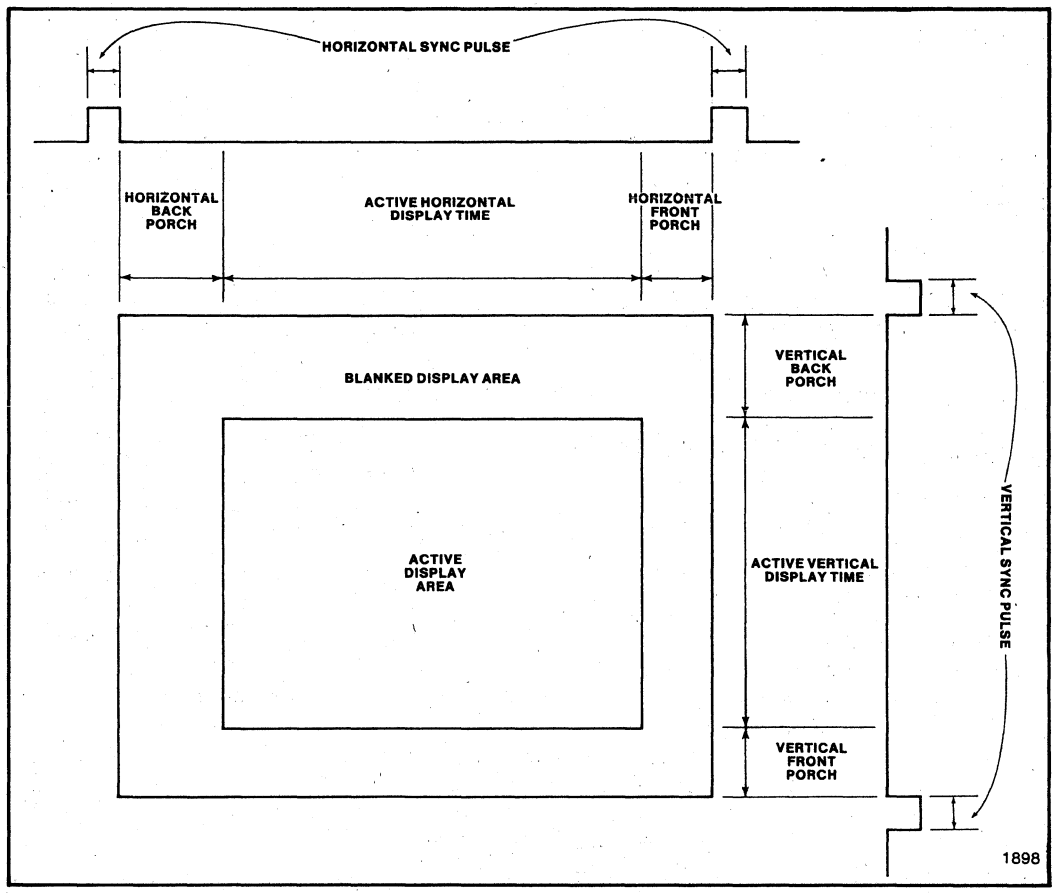


Figure 8. Monitor Timing

guarantees that the electron beam will be blanked by the time the sync pulse arrives. During the sync pulse blanking time, the electron gun retraces back to the left and/or top of the CRT. Following retrace, a discrete time period is required to position the electron gun at the first pixel of the new raster line. Therefore, a monitor also specifies minimum horizontal and vertical back porch blanking times, which immediately follow the corresponding sync pulse. Thus, horizontal or vertical blanking time is defined by the respective front porch, sync pulse, and back proch. The iSBX 275 board does not input any display memory bits to the electron gun during blanking time.

Fig. 8 shows the relationship of active display time and blanking time to the entire raster-scan area. Increasing the horizontal front porch, sync, or back porch decreases the number of pixels which are displayed in the horizontal direction. Increasing the vertical values decreases the number of lines which are displayed. These values should therefore be set such that the active display area exactly matches the desired bit map configuration. The active display area then contains only those pixels which are mapped from the display memory on the iSBX 275 board.

INTERLACING

The horizontal resolution of a monitor, or the number of pixels in each display line, is a function of the bandwidth and the horizontal rate of the monitor, as described above. Given the monitor horizontal rate, a higher bandwidth means more pixels per line. The iSBX 275 board, with the 12.6 MHz default oscillator, provides a choice of two bandwidths, one twice the rate of the other. The slower rate is used for non-interlaced display mode operation. When interlaced display mode operation is selected, doubling the number of lines per inch, the bandwidth must also be doubled.

The vertical resolution of a monitor, or the number of lines displayed, is a function of the field rate and the horizontal rate of the monitor. Given these two rates, the vertical resolution is fixed. It can be increased then only through interlacing, which doubles the vertical resolution.

In non-interlaced display mode operation, the electron gun always begins and ends a raster-scan cycle in the lower right corner of the CRT, as shown in Fig. 6. A frame is thus comprised of one field, as each field is scanned directly over the preceding one. In interlaced display mode operation, a frame is comprised of two interlaced fields, as shown in Fig. 7. The electron gun always begins the first field and ends the second field in the lower right corner of the CRT. This is accomplished by scanning an odd number of lines for each frame. Since an odd number of lines cannot be equally divided between two fields, each field receives part of the extra line. The first field (Fig. 7a) begins its vertical retrace in the lower right corner and ends its vertical scan partway through the odd horizontal scan. The second field (Fig. 7b) then starts its vertical retrace partway through the odd horizontal scan and ends its vertical scan in the lower right corner. Due to the slanted nature of the scan lines, the two fields are vertically offset from each other by half of a line. (Refer to Fig. 7c.) Interlacing doubles the vertical resolution of the monitor by scanning twice as many lines per inch, but the frame rate is half of the non-interlaced display mode frame rate.

PL/M-86 GRAPHICS LIBRARY

Introduction

The graphics procedures described below provide a simple interface to the basic capabilities of the iSBX 275 board. These procedures require an iSBC® 337 Numeric Data Processor MULTIMODULE board or equivalent in the system. For this application note, the graphics procedures were placed into a library with the pathname /user/sbx275/lib/sbx275. Each module calling any of these procedures contained the following include file:

```
$include(/user/sbx275/ext/sbx275)
```

This include file contains all of the external declarations of the graphics procedures, as well as the literal declarations for all of the "parameter choices" available to the procedures. These "parameter choices" are listed below, as well as in the appropriate procedure descriptions.

```
zeros, ones
bw, red, blue, magenta, green, yellow, cyan,
  white
south, southeast, east, northeast, north,
  northwest, west, southwest
bw_plane, red_plane, blue_plane, green_plane
normal_char, slant_char
replace, complement, reset, set
one_x, two_x, three_x, four_x, five_x, six_x,
  seven_x, eight_x, nine_x, ten_x,
  eleven_x, twelve_x, thirteen_x, fourteen_x,
  fifteen_x, sixteen_x
```

"Parameter Choice" Literals

For addressing purposes, the monitor screen is in the first quadrant of the Cartesian plane. The origin is in the lower left corner of the display, with positive x to the right and positive y up.

Overview

First, the graphics procedures are divided into several functional groups and briefly defined, with those in each group listed alphabetically. This provides an easy reference to the procedures rather than a detailed description of them. Then all of the parameters used by these procedures are defined, in alphabetical order. The parameter definitions provide a more extensive description of the capabilities of these procedures. Except for those parameters defined as pointers or byte variables, all parameters are 16-bit word values.

Initialization Procedures

Of the following routines, only INIT must be called by the applications programmer. The rest are called as a result of the INIT call.

ABCMD: causes the GDC to exit idle mode and enable (begin) the display.

ABCMD must be called following an ARCMD call, in order to unblank the display.

Calls: CMDOUT.

Called by UICMD.

ACCMD: disables the cursor.

Calls: CMDOUT, DATOUT.

Called by UICMD.

ARCMD(reset_parameters_pointer): resets the GDC and puts it in idle mode, blanking the display.

Calls: CMDOUT, DATOUT.

Called by UICMD.

AVCMD(1): sets the GDC as the master vertical sync generator.

Calls: CMDOUT.

Called by UICMD.

INIT(plane_size,xres,yres,hs,vs,hfp,hbp,vfp, vbp,opmode,command_port,data_port):

assigns the bit map, monitor, and operation mode parameters;
assigns the iSBX 275 board I/O port addresses;
calls UICMD, initializing the iSBX 275 board;
calls UPCMD and AWCMD, clearing display memory;
calls AGCMD, initializing for solid area fill and line drawing;
calls UMCMD, setting the drawing color to BW and the drawing mode to SET.

INIT must be the first iSBX 275 board procedure called by a user program. It need only be called once for a given application program.

UICMD: calls INITFP from 8087.LIB, initializing the 8087;
calls ARCMD, resetting the GDC and putting it in idle mode;
calls APCMD, setting the pitch equal to the display area width;
calls AZCMD, setting the zoom magnification factor to ONE_X;
calls ACCMD, disabling the cursor display;
calls ASCMD, setting up a single display window which starts at address 0 and comprises the entire display area;
calls AVCMD, setting the GDC as the master video sync generator;
calls ABCMD, beginning the display.

Called by INIT.

Control Procedures

These routines provide control for drawing in the bit map.

AGCMD(pattern7_8,pattern5_6,pattern3_4, pattern1_2): loads an 8 × 8 bit drawing pattern.

INIT loads a pattern for solid area fill, which also defines solid line drawing.

Calls: CMDOUT, DATOUT.

Called by INIT.

APCMD(words): sets the pitch (bit map width).

INIT sets the pitch equal to the display area width, which is (xres÷16) display memory words.

In black-and-white, non-interlaced display mode, the bit map typically will be larger than the display area. With the pitch equal to (xres÷16), the bit map will be longer than the display area. If an APCMD call sets the pitch greater than (xres÷16), the bit map will be at least wider and possibly also longer than the display area. Panning and scrolling of the display windows over this larger bit map are accomplished with ASCMD.

Calls: CMDOUT, DATOUT.

Called by UICMD.

ASCMD(start_address1,length1,start_address2, length2): defines the top and bottom display windows into which the display area can be divided.

INIT sets up only one display window, which starts at address 0 and comprises the entire display area.

In black-and-white, non-interlaced display mode, ASCMD can be used to pan or scroll the display windows independently over the bit map. If an APCMD call has set the pitch wider than the display area, the display windows can be panned across the bit map. Panning is accomplished by a series of ASCMD calls which increment or decrement, by one, the starting address of a window. Panning a display window beyond the left or right borders of the bit map will distort the displayed image. If the bit map is longer than the display area, the display windows can be scrolled over the bit map. Scrolling is accomplished by a series of ASCMD calls which increment or decrement, by the pitch, the starting address of a window. Scrolling a display window beyond the top or bottom borders of the bit map may not be desireable.

Calls: CMDOUT, DATOUT.

Called by UICMD.

AWCMD (write_mode , direction , aw_pattern, aw_count): writes multiple words of all zeros or all ones directly to display memory.

An AWCMD call must be preceded by a UPCMD call. The following sequence will clear display memory:

```
call UPCMD(0,0,BW_PLANE);
call AWCMD ( REPLACE , EAST ,
                    ZEROS, 4000H);
```

Calls: CMDOUT, DATOUT.

Called by INIT.

AZCMD(zoom_factor): sets the drawing magnification factor.

INIT sets zoom_factor to ONE_X.

Calls: CMDOUT, DATOUT.

Called by UICMD, STRCMD.

UMCMD(color,write_mode): sets the drawing color and mode.

INIT sets color to BW and write_mode to SET.

Called by INIT, STRCMD.

UPCMD(x1,y1,plane): positions the cursor in display memory.

Calls: POSCUR.

Called by INIT.

Figure Drawing Procedures

These routines draw figures in the bit map.

UACMD(xc,yc,radius,direction,angle_T,angle_S): draws an arc.

Calls: CAP, CMDOUT, POSCUR, LPARMS, DFIG.

UBCMD(x1,y1,length,height,direction): draws a rectangle (box).

Calls: CMDOUT, POSCUR, LPARMS, DFIG.

UCCMD(xc,yc,radius): draws a circle.

Calls: CAP, CMDOUT, POSCUR, LPARMS, DFIG.

UFCMD(x1,y1,length,height,direction): fills a rectangular area with the pattern specified by the most recent AGCMD call.

Calls: CMDOUT, POSCUR, LPARMS.

ULCMD(x1,y1,x2,y2): draws a line.

Calls: CMDOUT, POSCUR, LPARMS, DFIG.

USCMD(x1,y1,direction): draws the symbol defined by the most recent AGCMD call.

Calls: CMDOUT, POSCUR, LPARMS.

Text Drawing Procedures

These routines draw ASCII characters in the bit map.

STRCMD(x1,y1,slant,direction,zoom_factor,color, text_length,text_pointer): draws an ASCII string.

An STRCMD call alters the fill and draw pattern, so AGCMD must be called to reload the pattern before resuming figure drawing.

An STRCMD call sets zoom_factor to ONE_X before returning.

Calls: AZCMD, UMCMD, UDCMD.

UDCMD(x1,y1,slant,direction,character): draws an ASCII character.

A UDCMD call alters the fill and draw pattern, so AGCMD must be called to reload the pattern before resuming figure drawing.

Calls: CMDOUT, DATOUT, POSCUR, LPARMS.

Support Procedures

None of these routines would generally be called by an applications programmer. They are called by the routines described in the previous procedure groups.

CAP(radius,angle_S,direction,angle_T,arc_parameter_pointer): calculates arc drawing parameters.

Calls: mqerSIN, mqerIE2 from CEL87.LIB.

Called by UACMD, UCCMD.

CMDOUT(command_byte): writes a command to the GDC command port and logs it in mem$buf, a 256-byte circular buffer.

Called by ABCMD, ACCMD, AGCMD, APCMD, ARCMD, ASCMD, AVCMD, AWCMD, AZCMD, UACMD, UBCMD, UCCMD, UDCMD, UFCMD, ULCMD, USCMD, DFIG, LPARMS, POSMP.

DATOUT(data_byte): writes a data byte to the GDC data port and logs it in mem$buf, a 256-byte circular buffer.

Called by ACCMD, AGCMD, APCMD, ARCMD, ASCMD, AWCMD, AZCMD, UDCMD, LPARMS, POSMP.

DFIG: starts the GDC drawing a figure.

Calls: CMDOUT.

Called by UACMD, UBCMD, UCCMD, ULCMD.

LPARMS(type_and_direction,dc,d,d2,d1,dm): loads the figure drawing parameters to the GDC.

Calls: CMDOUT, DATOUT.

Called by UACMD, UBCMD, UCCMD, UDCMD, UFCMD, ULCMD, USCMD.

POSCUR(x1,y1,plane): converts a cursor address given relative to the x-y coordinate system into an absolute display memory address, which it sends to the GDC via POSMP.

Calls: POSMP.

Called by UACMD, UBCMD, UCCMD, UDCMD, UFCMD, ULCMD, UPCMD, USCMD.

POSMP(plane,word_address,dot_address): loads an absolute cursor address to the GDC.

Calls: CMDOUT, DATOUT.

Called by POSCUR.

Character Generator

CG5X7: character font table for drawing 5 × 7 characters in an 8 × 8 matrix.

Used by UDCMD, STRCMD.

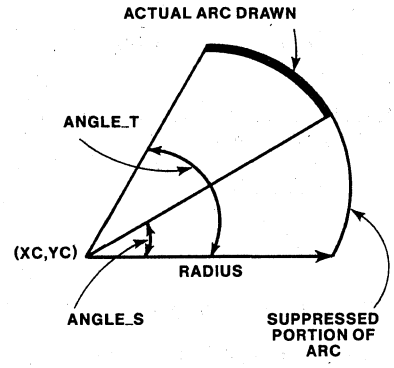
Parameter Definitions

The parameters defined here are used by the routines described briefly above.

angle_S: the angle from the major axis to the start of the arc, $0 \leq$ angle_S < 45, in degrees. (See the figure below.)

angle_T: the angle from the major axis to the end of the arc, $0 <$ angle_T ≤ 45, in degrees. (See the figure below.)

The major axis is the x or y axis with respect to which an arc angle is measured.



arc_parameters_pointer: a pointer to the table containing arc drawing parameters.

aw_count: the number of consecutive words AWCMD is to write to display memory.

aw_pattern: the 16-bit pattern AWCMD uses to write to display memory.

Parameter choices are ZEROS or ONES.

character: the ASCII code for the character to be drawn by USCMD.

color: the drawing color.

Parameter choices are BW, RED, BLUE, MAGENTA, GREEN, YELLOW, CYAN, or WHITE.

BW is white in black-and-white display mode.

Color Plane			Displayed Color
Green	Blue	Red	
0	0	0	Black
0	0	1	Red
0	1	0	Blue
0	1	1	Magenta
1	0	0	Green
1	0	1	Yellow
1	1	0	Cyan
1	1	1	White

command_byte: a byte which the GDC interprets as a command.

command_port: the address of the I/O port on the SBX baseboard through which commands will be sent to the iSBX 275 board.

The I/O port must support 8-bit MULTIMODULE™ boards. When addressed, the port must activate MCS0/ on the SBX bus.

data_byte: a parameter byte which the GDC interprets according to the relative order in which the GDC receives it, following the most recent command byte.

data_port: the address of the I/O port on the SBX baseboard through which data will be sent to the iSBX 275 board.

The I/O port must support 8-bit MULTIMODULE boards. When addressed, the port must activate MCS0/ on the SBX bus.

dc,d,d2,d1,dm: five figure drawing parameters which are interpreted by the GDC according to the drawing type, as described below.

DRAWING TYPE	DC	D	D2	D1	DM
INITIAL VALUE*	0	8	8	−1	−1
LINE	$\lvert\triangle\rvert$	$2\lvert\triangle D\rvert-\lvert\triangle\rvert$	$2(\lvert\triangle D\rvert-\lvert\triangle\rvert)$	$2\lvert\triangle D\rvert$	-
ARC**	$r\sin\phi\rfloor$	4−1	$2(r-1)$	−1	$r\sin\theta\rfloor$
RECTANGLE	3	A−1	B−1	−1	A−1
AREA FILL	B−1	A	A	-	-
GRAPHIC CHARACTER***	B−1	A	A	-	-
WRITE DATA	W−1	-	-	-	-
READ DATA	W	-	-	-	-

*INITIAL VALUES FOR THE VARIOUS PARAMETERS ARE LOADED WHEN THE FIGS COMMAND BYTE IS PROCESSED.
**CIRCLES ARE DRAWN WITH 8 ARCS, EACH OF WHICH SPAN 45°, SO THAT $\sin\phi = 1/\sqrt{2}$ AND $\sin\theta = 0$.
***GRAPHIC CHARACTERS ARE A SPECIAL CASE OF BIT-MAP AREA FILLING IN WHICH B AND A ≤ 8. IF A = 8 THERE IS NO NEED TO LOAD D AND D2.

WHERE:
−1 = ALL ONES VALUE.
ALL NUMBERS ARE SHOWN IN BASE 10 FOR CONVENIENCE. THE GDC ACCEPTS BASE 2 NUMBERS (2s COMPLEMENT NOTATION WHERE APPROPRIATE).
- = NO PARAMETER BYTES SENT TO GDC FOR THIS PARAMETER.
$\lvert\triangle\rvert$ = THE LARGER OF $\triangle x$ OR $\triangle y$.
$\triangle D$ = THE SMALLER OF $\triangle x$ OR $\triangle y$.
r = RADIUS OF CURVATURE, IN PIXELS.
ϕ = ANGLE FROM MAJOR AXIS TO END OF THE ARC. $\phi \leq 45°$.
θ = ANGLE FROM MAJOR AXIS TO START OF THE ARC. $\theta \leq 45°$.
$\lceil\ \rceil$ = ROUND UP TO THE NEXT HIGHER INTEGER.
$\lfloor\ \rfloor$ = ROUND DOWN TO THE NEXT LOWER INTEGER.
A = NUMBER OF PIXELS IN THE INITIALLY SPECIFIED DIRECTION.
B = NUMBER OF PIXELS IN THE DIRECTION AT RIGHT ANGLES TO THE INITIALLY SPECIFIED DIRECTION.
W = NUMBER OF WORDS TO BE ACCESSED.
DC = DRAWING COUNT PARAMETER WHICH IS ONE LESS THAN THE NUMBER OF RMW CYCLES TO BE EXECUTED.
DM = DOTS MASKED FROM DRAWING DURING ARC DRAWING.

direction: the initial drawing direction.

Parameter choices are SOUTH, SOUTHEAST, EAST, NORTHEAST, NORTH, NORTHWEST, WEST, or SOUTHWEST.

	DIR	LINE	ARC	CHARACTER	SLANT CHAR	RECTANGLE
SOUTH:	000					
SOUTHEAST:	001					
EAST:	010					
NORTHEAST:	011					
NORTH:	100					
NORTHWEST:	101					
WEST:	110					
SOUTHWEST:	111					

dot_address: the portion of the absolute cursor address which specifies the bit location within the display memory word specified by word_address.

hbp: the horizontal back porch, expressed in display memory words.

The hbp parameter is dependent on hs, hfp, and AW. (See the xres definition.)

Interlaced Display Mode

For interlaced display mode, both of the following equations must be satisfied, with hbp rounded up to the nearest integer:

hbp ≥ 5, a GDC requirement; and

hbp ≥ (monitor min. hbp) × (iSBX 275 board bandwidth) ÷ 16.

Non-interlaced Display Mode

For non-interlaced display mode, all three of the following equations must be satisfied, with hbp rounded up to the nearest integer:

hbp ≥ 5, a GDC requirement if two display windows are used:

hbp ≥ 3, a GDC requirement; and

hbp ≥ (monitor min. hbp) × (iSBX 275 board bandwidth) ÷ 16.

height: the number of pixels in the dimension perpendicular to the specified drawing direction.

hfp: the horizontal front porch, expressed in display memory words.

The hfp parameter is dependent on hs, hbp, and AW. (See the xres definition.)

Interlaced Display Mode

For interlaced display mode, the following three equations must be satisfied, with hfp rounded up to the nearest integer:

hfp ≥ 6, a GDC requirement if the light pen input is used;

hfp ≥ 2;

hfp ≥ (monitor min. hfp) × (iSBX 275 board bandwidth) ÷ 16.

Non-interlaced Display Mode

For non-interlaced display mode, the following three equations must be satisfied, with hfp rounded up to the nearest integer:

hfp ≥ 6, a GDC requirement if the light pen input is used;

hfp ≥ 2;

hfp ≥ (monitor min. hfp) × (iSBX 275 board bandwidth) ÷ 16.

hs: the horizontal sync, expressed in display memory words.

The hs parameter is dependent on hfp, hbp, and AW. (See the xres definition.)

Interlaced Display Mode

For interlaced display mode, both of the following equations must be satisfied, with hs rounded up to the nearest integer:

hs ≥ 5, a GDC requirement; and

hs ≥ (monitor min. hs) × (iSBX 275 board bandwidth) ÷ 16.

Non-interlaced Display Mode

For non-interlaced display mode, the following equation must be satisfied, with hs rounded up to the nearest integer:

hs ≥ (monitor min. hs) × (iSBX 275 board bandwidth) ÷ 16.

length: the number of pixels in the specified drawing direction.

length1: the number of display lines comprising the top display window.

length2: the number of display lines comprising the bottom display window.

opmode: the operation mode of the GDC.

MSB | 0 0 0 F I 0 G S | LSB

I	S	Description
0	0	Non-interlaced
0	1	Invalid
1	0	Interlaced Repeat Field for Character Displays
1	1	Interlaced

G	Description
0	Mixed Graphics and Character Mode
1	Graphics Mode

F	Description
0	Drawing During Active Display Time and Retrace Blanking
1	Drawing Only During Retrace Blanking

pattern7_8,pattern5_6,pattern3_4,pattern1_2: the four pieces of an 8 × 8 bit pattern, used for drawing a random 8 × 8 symbol or filling a rectangular area.

The pattern is applied backwards, as shown in the example below. The example assumes that write_mode is set to REPLACE.

For pattern7_8 = 0602h,
pattern5_6 = 0a0eh,
pattern3_4 = 3212h,
pattern1_2 = 003eh,

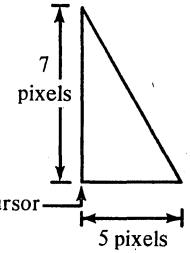
the 8 × 8 bit pattern and the resulting figure displayed are as shown. The initial cursor position is designated by the arrows.

8 × 8 BIT PATTERN

0	0	0	0	0	0	1	0	8
0	0	0	0	0	1	1	0	7
0	0	0	0	1	1	1	0	6
0	0	0	0	1	0	1	0	5
0	0	0	1	0	0	1	0	4
0	0	1	1	0	0	1	0	3
0	0	1	1	1	1	1	0	2
0	0	0	0	0	0	0	0	1

cursor

FIGURE DISPLAYED

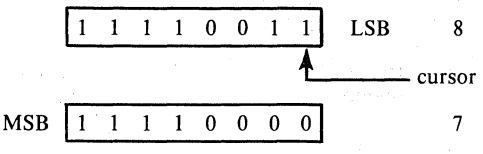


For each line, rectangle, arc, or circle drawing command issued, pattern7_8 also serves as the 16-bit drawing pattern. For each such command issued, pattern7_8 is applied repetitively from least- to most-significant bit, as shown in the example below. The example assumes that write_mode is set to REPLACE.

For pattern7_8 = 0f0f3h,

the 2 × 8 bit portion of the 8 × 8 bit pattern, and the resulting line pattern that is drawn, are as shown. The initial cursor position is also shown.

PATTERN7_8



LINE PATTERN

plane: the color plane in which drawing is to occur. (See the color parameter definition.)

Parameter choices are BW_PLANE, RED_PLANE, BLUE_PLANE, or GREEN_PLANE.

- BW_PLANE - starts at display memory word address 0 (for black-and-white display mode).
- RED_PLANE - starts at display memory word address 0.
- BLUE_PLANE - starts at display memory word address 1000h.
- GREEN_PLANE - starts at display memory word address 2000h.

Display memory word addresses 3000h to 3ffffh are not used in color mode.

plane_size: the number of display memory words per plane.

For black-and-white display mode, the entire display memory is used as a single-plane bit map, so

plane_size = 16,384 (or 4000h).

For color display mode, display memory is divided into four equal planes, one for each of the three primary colors and one which is unused, so

plane_size = 4096 (or 1000h).

radius: the distance, in pixels, from the center of an arc or circle to any point on that arc or circle.

reset_parameters_pointer: a pointer to the table in INIT containing the monitor and operation mode parameters for ARCMD.

slant: the character style. (See the direction parameter definition.)

Parameter choices are NORMAL_CHAR or SLANT_CHAR.

start_address1: the address of the first display memory word of the top display window.

start_address2: the address of the first display memory word of the bottom display window.

text_length: the number of ASCII bytes to be taken from the text buffer by STRCMD and drawn in the bit map.

text_pointer: a pointer to the text buffer used by STRCMD.

type_and_direction: the drawing direction and type.

MSB | S | R | A | G | L | DIR | LSB

	Drawing Type
S	Slanted Graphics Character
R	Rectangle
A	Arc/Circle
G	Graphics Character
L	Line (Vector)

A "1" enables a type.

DIR - see direction description

vbp: the vertical back porch, expressed in display lines.

The vbp parameter is dependent on vs, vfp, and AL. (See the yres definition.)

The following equation must be satisfied, with vbp rounded up to the nearest integer:

$vbp \geq$ (monitor min. vbp) × (monitor horizontal rate).

vfp: the vertical front porch, expressed in display lines.

The vfp parameter is dependent on vs, vbp, and AL. (See the yres definition.)

The following equation must be satisfied, with vfp rounded up to the nearest integer:

$vfp \geq$ (monitor min. vfp) × (monitor horizontal rate).

vs: the vertical sync, expressed in display lines.

The vs parameter is dependent on vfp, vbp, and AL. (See the yres definition.)

The following equation must be satisfied, with vs rounded up to the nearest integer:

$vs \geq$ (monitor min. vs) × (monitor horizontal rate).

words: the pitch (width of the bit map), expressed in display memory words.

word_address: the portion of the absolute cursor address which specifies the display memory word location in the color plane specified by the plane parameter.

write_mode: determines how the drawing pattern, specified by the most recent AGCMD call (or by aw_pattern for AWCMD), is used to modify bits in display memory.

Parameter choices are REPLACE, COMPLEMENT, RESET, or SET.

COMPLEMENT - drawing commands
RESET complement, reset,
SET or set, respectively, only those bits in display memory which correspond to ones in the drawing pattern.

REPLACE - the same as SET, except that drawing commands also reset the bits in display memory which correspond to zeros in the drawing pattern.

x1,y1: the beginning coordinates of a line, rectangle, rectangular area to be filled, or character.

x2,y2: the ending coordinates of a line.

xc,yc: the center coordinates of an arc or circle.

xres: the horizontal resolution, or the number of pixels displayed in the x direction.

The xres parameter is dependent on hs, hfp, and hbp. The following procedure for determining these four parameters makes the most efficient use of the bit map on the iSBX 275 board. AW is the number of active display words (xres÷16) per raster line. The GDC requires that AW be an even integer.

Interlaced Display Mode or Black-and-white, Non-interlaced Display Mode

In interlaced display mode (or in black-and-white, non-interlaced display mode), the horizontal timing parameters must satisfy the equation below (Eq. 1x). AW is an even integer, and the quotient on the right must be rounded down to the nearest integer:

$$hs + hfp + hbp + AW = \frac{\text{iSBX 275 board bandwidth}}{16 \times (\text{monitor horizontal rate})}$$

(Eq. 1x)

The first step is to find AW. The maximum possible resolution supported by the iSBX 275 board in interlaced display mode (or in black-and-white, non-interlaced display mode) is

$$\text{resolution} \leq 512 \times 512$$

Given the desired monitor aspect ratio, the bit map must be configured as a grid with the same x:y ratio, such that

$$\text{resolution} = x \times y, \text{ or}$$

$$512 \times 512 \geq (AW \times 16) \times \frac{AW \times 16}{\text{aspect ratio}}$$

(Eq. 2x)

Using Eq. 3x below, which is derived from Eq. 2x, calculate the maximum value for AW, rounding down to the nearest even integer:

$$AW \leq 32 \times \sqrt{\text{aspect ratio}}$$

(Eq. 3x)

The next step is to calculate the minimum values for hs, hfp, and hbp. (Refer to the respective parameter definitions.) Then adjust the values of AW, hs, hfp, and hbp, as required, to satisfy Eq. 1x above. Finally, solve for xres using Eq. 4x below:

$$xres = 16 \times AW$$

(Eq. 4x)

Color, Non-interlaced Display Mode

In color, non-interlaced display mode, the horizontal timing parameters must satisfy the equation below (Eq. 5x). AW is an even integer, and the quotient on the right must be rounded down to the nearest integer:

$$hs + hfp + hbp + AW = \frac{\text{iSBX 275 board bandwidth}}{16 \times (\text{monitor horizontal rate})}$$

(Eq. 5x)

The first step is to find AW. The maximum possible resolution supported by the iSBX 275 board in color, non-interlaced display mode is

$$\text{resolution} \leq 256 \times 256$$

Given the desired monitor aspect ratio, the bit map must be configured as a grid with the same x:y ratio, such that

$$\text{resolution} = x \times y, \text{ or}$$

$$256 \times 256 \geq (AW \times 16) \times \frac{AW \times 16}{\text{aspect ratio}}$$

(Eq. 6x)

Using Eq. 7x below, which is derived from Eq. 6x, calculate the maximum value for AW, rounding down to the nearest even integer:

$$AW \leq 16 \times \sqrt{\text{aspect ratio}}$$

(Eq. 7x)

The next step is to calculate the minimum values for hs, hfp, and hbp. (Refer to the respective parameter definitions.) Then adjust the values of AW, hs, hfp, and hbp, as required, to satisfy Eq. 5x above. Finally, solve for xres using Eq. 8x below:

$$xres = 16 \times AW$$

(Eq. 8x)

yres: the vertical resolution, or the number of lines displayed.

The yres parameter is dependent on vs, vfp, and vbp. The following procedure for determining these four parameters makes the most efficient use of the bit map on the iSBX 275 board. AL is the number of active display lines per field.

Interlaced Display Mode

In interlaced display mode, the vertical timing parameters must satisfy the equation below (Eq. 1y). AL is an even integer, and the quotient on the right must be rounded down to the nearest even integer:

$$vs + vfp + vbp + AL = \frac{\text{monitor horizontal rate}}{\text{monitor field rate}} \qquad \textit{(Eq. 1y)}$$

The first step is to find AL. The maximum possible resolution supported by the iSBX 275 board in interlaced display mode is

$$\text{resolution} \leq 512 \times 512$$

Only half of the active display lines are contained in each field in interlaced display mode, so given xres,

$$\text{resolution} = x \times (2 \times y), \text{ or}$$

$$512 \times 512 \geq \text{xres} \times (2 \times AL) \qquad \textit{(Eq. 2y)}$$

Using Eq. 3y below, which is derived from Eq. 2y, calculate the maximum value for AL, rounding down to the nearest even integer:

$$AL \leq \frac{512 \times 512}{2 \times \text{xres}} \qquad \textit{(Eq. 3y)}$$

The next step is to calculate the minimum values for vs, vfp, and vbp. (Refer to the respective parameter definitions.) Then adjust the values of AL, vs, vfp, and vbp, as required, to satisfy Eq. 1y above. Finally, solve for yres using Eq. 4y below:

$$\text{yres} = 2 \times AL \qquad \textit{(Eq. 4y)}$$

Color, Non-interlaced Display Mode

In color, non-interlaced display mode, the vertical timing parameters must satisfy the equation below (Eq. 5y), in which the quotient on the right must be rounded down to the nearest integer:

$$vs + vfp + vbp + AL = \frac{\text{monitor horizontal rate}}{\text{monitor field rate}} \qquad \textit{(Eq. 5y)}$$

The first step is to find AL. The maximum possible resolution supported by the iSBX 275 board in color, non-interlaced display mode is

$$\text{resolution} \leq 256 \times 256$$

All of the active display lines are contained in each field in non-interlaced display mode, so given xres,

$$\text{resolution} = x \times y, \text{ or}$$

$$256 \times 256 \geq \text{xres} \times AL \qquad \textit{(Eq. 6y)}$$

Using Eq. 7y below, which is derived from Eq. 6y, calculate the maximum value for AL, rounding down to the nearest integer:

$$AL \leq \frac{256 \times 256}{\text{xres}} \qquad \textit{(Eq. 7y)}$$

The next step is to calculate the minimum values for vs, vfp, and vbp. (Refer to the respective parameter definitions.) Then adjust the values of AL, vs, vfp, and vbp, as required, to satisfy Eq. 5y above. Finally, solve for yres using Eq. 8y below:

$$\text{yres} = AL \qquad \textit{(Eq. 8y)}$$

Black-and-white, Non-interlaced Display Mode

In black-and-white, non-interlaced display mode, the vertical timing parameters must satisfy the equation below (Eq. 9y), in which the quotient on the right must be rounded down to the nearest integer:

$$vs + vfp + vbp + AL = \frac{\text{monitor horizontal rate}}{\text{monitor field rate}} \qquad \textit{(Eq. 9y)}$$

The first step is to find AL. The maximum possible resolution supported by the iSBX 275 board in black-and-white, non-interlaced display mode is

$$\text{resolution} \leq 512 \times 512$$

All of the active display lines are contained in each field in non-interlaced display mode, so given xres,

$$\text{resolution} = x \times y, \text{ or}$$

$$512 \times 512 \geq \text{xres} \times \text{AL} \qquad \textit{(Eq. 10y)}$$

Using Eq. 11y below, which is derived from Eq. 10y, calculate the maximum value for AL, rounding down to the nearest integer:

$$\text{AL} \leq \frac{512 \times 512}{\text{xres}} \qquad \textit{(Eq. 11y)}$$

The next step is to calculate the minimum values for vs, vfp, and vbp. (Refer to the respective parameter definitions.) Then adjust the values of AL, vs, vfp, and vbp, as required, to satisfy Eq. 9y above. Finally, solve for yres using Eq. 12y below:

$$\text{yres} = \text{AL} \qquad \textit{(Eq. 12y)}$$

zoom_factor: the factor by which a specified figure or character is magnified at the time it is drawn in the bit map.

Parameter choices are ONE_X, TWO_X, THREE_X, FOUR_X, FIVE_X, SIX_X, SEVEN_X, EIGHT_X, NINE_X, TEN_X, ELEVEN_X, TWELVE_X, THIRTEEN_X, FOURTEEN_X, FIFTEEN_X, or SIXTEEN_X.

APPLICATION EXAMPLES

Introduction

The application examples described below were developed and run on a SYSTEM 86/330A, which is an iRMX 86 based system containing the following:

- an iSBC 86/30 processor board with 128KB of onboard RAM
- an iSBC 337 Numeric Data Processor MULTIMODULE board, mounted on the iSBC 86/30 board
- an iSBC 56A 256KB RAM board
- an iSBC 215 Winchester disk controller board
- an iSBX 218 flexible disk controller MULTIMODULE board, mounted on the iSBC 215 board
- a 35MB Winchester disk drive
- an 8" flexible disk drive

The system ran the pre-configured version of iRMX 86, Rel. 5, and the application examples were written in PL/M-86.

The iSBX 275 board was connected to SBX port J4 in the center of the iSBC 86/30 board. A Mitsubishi C-3419 RGB color monitor was used for the display.

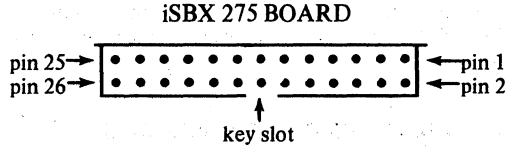
Overview

First, instructions are provided for building a cable to interface the iSBX 275 board to an RGB color monitor. Next, the AIRPLANE application is discussed, providing an example of black-and-white, interlaced display mode operation. Finally, the REACTOR application is discussed, providing an example of color display mode operation.

iSBX™ 275 Board Color Monitor Cable Fabrication

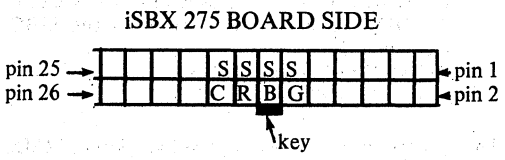
Since a BNC color monitor was chosen, a female BNC connector, Amphenol part number 69475, was first attached to each of four six-foot lengths of 75 ohm coax cable, Belden part number 83264-001. The four connectors were then labeled Red, Green, Blue, and Csync, corresponding to the respective male BNC connectors on the back of the monitor. Two crimp-on connector pins, BERG part number 75691-014, were then crimped and soldered to the other end of each cable piece, one to the wire and the other to the shielding. Finally, the four cable pieces were attached to a wire-type, keyed, 26-pin female connector, BERG part number 65846-819, as described below.

The keyed, 26-pin male connector on the iSBX 275 board is depicted here as viewed from above.



The 26-pin female connector for the cable is also depicted here as viewed from above (when inserted on the board connector). The crimp pins on the four cable pieces were inserted into this connector from the top. The figure shows the locations for inserting

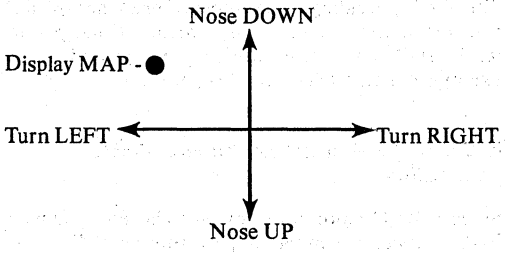
all eight of the Red(R), Green(G), Blue(B), Csync(C), and Shielding(S) crimp pins.



Airplane Application

INTRODUCTION

This application example is a simulation of a Cessna 152. For this example, the iSBX 275 board is configured for black-and-white, interlaced display mode operation. An Atari joystick, oriented such that the button is in the upper left corner, controls the "airplane" as shown.



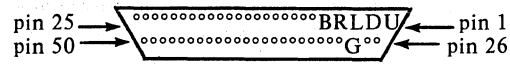
A real-time "flight panel" on the graphics monitor (see Fig. 9) reflects the "movement" of the plane as it is input by the joystick. When the joystick button is depressed and held, a map replaces the flight panel on the monitor. This map depicts the ground track of the plane and the surrounding area.

OVERVIEW

First, instructions are provided for building a cable to interface an Atari joystick to the iSBC 86/30 board. Then the necessary jumpers on the iSBC 86/30 board and the iSBX 275 board are detailed. Next, the process is described for determining the parameters for the INIT routine. Finally, the generation of the flight panel display shown in Fig. 9 is discussed.

ATARI JOYSTICK CABLE FABRICATION

A cable was built to connect an Atari joystick to the printer port 50-pin connector on the back of the SYSTEM 86/330A chassis. This enabled the joystick to communicate with the system via the parallel port on the iSBC 86/30 board. First, six three-foot lengths of 22 AWG wire were soldered to a 50-pin male connector, Ansley part number PN 609-50M. The figure below shows the locations for soldering the six wires to the connector, depicted here as viewed from the wire side. The letters represent the functions (U)p, (D)own, (L)eft, (R)ight, (B)utton, and (G)round.



Then the six wires were soldered to a 9-pin male connector. The figure below shows the locations for soldering the wires to the connector, depicted here as viewed from the wire side.

JUMPERS

A jumper was installed on the iSBC 86/30 board between posts 44 and 53. This made Port C, bit 7, available over the parallel port of the board.

This application required the maximum available resolution of the iSBX 275 board. The board was therefore jumpered for black-and-white display mode operation, resulting in a nominal resolution of 512 × 512. The board was also jumpered to provide an active low combined sync (CSYNC/) signal for the color monitor. The black-and-white display mode was selected by removing jumper 12 - 13. All of the installed jumpers are described below.

1 - 2 test jumper
8 - 10 provides CSYNC/ output
14 - 15 test jumper
16 - 17 selects non-divided clock

GRAPHICS INITIALIZATION

Introduction

In order to use the graphics library of PL/M-86 routines described earlier, the following include file was placed in the PLANE module:

```
$include(/user/sbx275/ext/sbx275)
```

An application program must call the procedure INIT before it calls any other procedure from this graphics library. Typically, INIT is called in the main procedure of an application, as was done in PLANE. (Refer to Fig. 10.) The iSBX 275 board is initialized according to the parameters accompanying the INIT

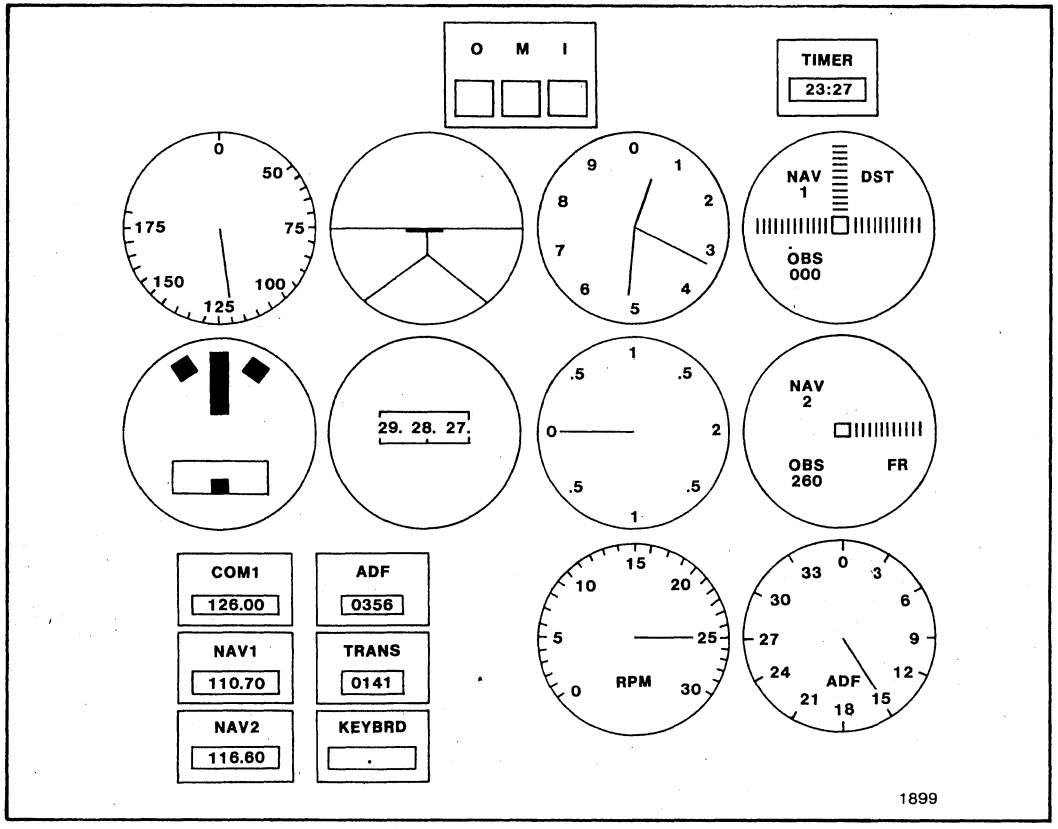


Figure 9. Flight Panel Display from Plane

call. These parameters specify the desired resolution, monitor, and mode of operation. They also specify the I/O port addresses of the iSBX 275 board. Following the INIT call, an application program may begin drawing by calling any of the other graphics procedures.

Overview

Many steps are involved in determining the parameters for INIT. First, the number and size of the bit map planes must be specified, providing the nominal resolution. Then the exact horizontal resolution and the associated monitor timing parameters must be determined. Next, the vertical resolution and the associated monitor timing parameters must be determined. Finally, the mode of operation and I/O addresses of the iSBX 275 board must be specified. The INIT parameters for PLANE were determined as described below.

Color Plane Size

The airplane application requires the black-and-white display mode of the iSBX 275 board. Referring to the description of the plane_size parameter, we find for this mode that

plane_size = 16384 (or 4000h).

Horizontal Resolution

The airplane application requires the interlaced display mode of the monitor. Referring to the description of the xres parameter, we find that the horizontal timing parameters must satisfy Eq. 1x. Given the 12.6 MHz oscillator shipped on the iSBX 275 board and the monitor horizontal rate of 15.75 KHz, Eq. 1x becomes

$$hs + hfp + hbp + AW = \frac{12.6\ \text{MHz}}{16 \times 15.75\ \text{KHz}}$$

$$= 50, \qquad \textit{(Eq. P1)}$$

which is an integer value as required.

```
*************************************************
                    MAIN
*************************************************

 1219   test: procedure public;

 1220   declare (x1,x2,y1,y2) word;

 1221   declare adc$use byte;
 1222   declare mkr$test byte;
 1223   declare trim real;
 1224   declare (low$byte, high$byte) word;

        /* initialize controller */
 1225   call init (16384,576,452,5,4,4,5,16,16,
                                  1bh,82h,80h);
        /* initial frequencies */
 1226   call movb (@initial$com(0),
                                  @com$freq(0),5);
 1227   call movb (@initial$nav1(0),
                                  @nav1$freq(0),5);
 1228   call movb (@initial$nav2(0),
                                  @nav2$freq(0),5);
 1229   call movb (@initial$adf(0),
                                  @adf$freq(0),4);
 1230   call movb (@initial$tran(0),
                                  @tran$freq(0),4);
```

Figure 10. INIT Call In PLANE

Still referring to the xres parameter description, we find that the maximum value for AW is determined using Eq. 3x. A 4:3 aspect ratio is desired, so given the nominal 255mm:190mm aspect ratio of the monitor, Eq. 3x becomes

$$AW \leqslant 32 \times \sqrt{255\text{mm} \div 90\text{mm}} = 37.1,$$

which, when rounded down as required to the nearest even integer, becomes

$$AW \leqslant 36.$$

The monitor manual specifies a minimum horizontal sync of 4.4 μs. Referring to the description of the HS parameter, we find that the minimum value for hs must be the greater of '5' and

$$hs \geqslant 4.4\ \mu s \times 12.6\ \text{MHz} \div 16 = 3.47,$$

and since $5 > 3.47$,

$$hs \geqslant 5.$$

The monitor manual specifies a minimum horizontal front porch of 1.2μs. Referring to the description of the hfp parameter, we find that the minimum value for hfp must be the greater of '2' and

$$hfp \geqslant 1.2\ \mu s \times 12.6\ \text{MHz} \div 16 = 0.945,$$

and since $2 > 0.945$,

$$hfp \geqslant 2.$$

The monitor manual specifies a minimum horizontal back porch of 4.9 μs. Referring to the description of the hbp parameter, we find that the minimum value for hbp must be the greater of '5' and

$$hbp \geqslant 4.9\ \mu s \times 12.6\ \text{MHz} \div 16 = 3.86,$$

and since $5 > 3.86$,

$$hbp \geqslant 5.$$

Using the values determined above,

$$hs + hfp + hbp + AW = 5 + 2 + 5 + 36 = 48,$$

which is two less than required by Eq. P1 above. The hs, hfp, and hbp values determined above are minimums, while the AW value is a maximum. Since the sum of these values must be incremented by two to satisfy Eq. P1, the hfp parameter is arbitrarily selected to be incremented by two. The horizontal parameter values are thus

$$AW = 36,\ hs = 5,\ hfp = 4,\ hbp = 5.$$

Referring once more to the description of the xres parameter, we find from Eq. 4x that

$$xres = 16 \times 36 = 576.$$

Vertical Resolution

Referring to the description of the yres parameter, we find that the vertical timing parameters must satisfy Eq. 1y. Given the monitor field rate of 60 Hz, Eq. 1y becomes

$$vs + vfp + vbp + AL = 15.75\ \text{KHz} \div 60\ \text{Hz} = 262.5,$$

which, when rounded down as required to the nearest even integer, becomes

$$vs + vfp + vbp + AL = 262. \qquad \textit{(Eq. P2)}$$

Still referring to the yres parameter description, we find that the maximum value for AL is determined using Eq. 3y, which becomes

$$AL \leqslant (512 \times 512) \div (2 \times 576) = 227.6,$$

which, when rounded down as required to the nearest even integer, becomes

$$AL \leqslant 226.$$

The monitor manual specifies a minimum vertical sync of three horizontal retrace-and-scan periods. Referring to the description of the vs parameter, we find that the minimum value for vs must be

$$vs \geq (3 \div 15.75\ KHz) \times 15.75\ KHz = 3.$$

The monitor manual specifies a minimum vertical front porch of three horizontal retrace-and-scan periods. Referring to the description of the vfp parameter, we find that the minimum value for vfp must be

$$vfp \geq (3 \div 15.75\ KHz) \times 15.75\ KHz = 3.$$

The monitor manual specifies a minimum vertical back porch of twelve horizontal retrace-and-scan periods. Referring to the description of the vbp parameter, we find that the minimum value for vbp must be

$$vbp \geq (12 \div 15.75\ KHz) \times 15.75\ KHz = 12.$$

Using the values determined above,

$$vs+vfp+vbp+AL=3+3+12+226=244,$$

which is eighteen less than required by Eq. P2 above. The vs, vfp, and vbp values determined above are minimums, while the AL value is a maximum. Since the sum of these values must be incremented by eighteen to satisfy Eq. P2, the vs, vfp, and vbp parameters are arbitrarily selected to be incremented by 1, 13, and 4, respectively. The vertical parameter values are thus

$$AL = 226,\ vs = 4,\ vfp = 16,\ vbp = 16.$$

Referring once more to the description of the yres parameter, we find from Eq. 4y that

$$yres = 2 \times 226 = 452.$$

Operation Mode and I/O Addresses

Referring to the description of the opmode parameter, we choose

opmode = 1BH.

This initializes the GDC for interlaced display mode operation; for individual pixel addressing and drawing; and for drawing in display memory only during horizontal and vertical retrace, when display memory is blanked from the monitor.

The iSBX 275 board resides on SBX port J4 of the iSBC 86/30 board. Referring to the description of the command_port and data_port parameters, we find that the I/O address chosen for each of these parameters must activate the MCS0/ signal on the SBX bus. Referring then to the iSBC 86/14 AND iSBC 86/30 SINGLE BOARD COMPUTER HARDWARE REFERENCE MANUAL, Intel order number 144044-002, page 3-7, we find that any of the I/O addresses 80, 82, 84, 86, 88, 8A, 8C, or 8E will activate MCS0/. Finally, referring to the iSBX 275 VIDEO GRAPHICS CONTROLLER MULTIMODULE BOARD REFERENCE MANUAL, Intel order number 144829-001, page 3-1, Table 3-1, we find that any of the I/O addresses 82, 86, 8A, or 8E may be used for the command_port parameter, and any of the I/O addresses 80, 84, 88, or 8C may be used for the data_port parameter. The I/O addresses chosen are therefore

command_port = 82H,

data_port = 80H.

CODE GENERATION

PLANE generates two different bit map displays, one for the flight panel and one for the ground track. Based on inputs from a joystick, these displays are updated in the bit map and displayed on the monitor screen. This section addresses only the flight panel display.

Before any code was generated, the flight panel was drawn on grid paper. This paper was layed out according to the xres and yres parameter values determined above for the INIT routine. The paper had 576 squares in the horizontal direction and 452 squares in the vertical direction, so each square represented a pixel on the display. The lower left square was assigned the (x,y) coordinates (0,0), with x increasing to the right and y increasing up.

The DRAW$PANEL procedure was then generated. This procedure draws the various flight panel elements in display memory, according to their size and location on the grid paper. (Refer to Fig. 9.) DRAW$PANEL first draws the six control boxes in the lower left corner of the flight panel display. Then the marker and timer boxes at the top of the flight panel display are drawn. Finally, the stationary portions of the ten indicators are drawn. The airspeed indicator in the upper left corner is drawn first. Then the turn-and-bank indicator, directly below the airspeed indicator, is drawn. Next, the artificial horizon indicator, directly to the right of the airspeed indicator, is drawn. The remaining seven indicators are drawn in the same fashion, progressing from top to bottom and from left to right. The dynamic portions of the flight panel display are drawn by the update procedures for the various controls and indicators.

Since PLANE generates two different bit map displays, DRAW$PANEL first clears display memory. (Refer to the listing of DRAW$PANEL in Appendix A.1.) It does this by calling UPCMD and AWCMD as explained in the AWCMD description. Then, since the drawing pattern is in an unknown state at the invocation of DRAW$PANEL, AGCMD is called to set the drawing pattern for solid line drawing and area fill. All lines subsequently drawn will thus be solid, and all areas filled will be solid filled. Next, UMCMD is called to change the write mode from REPLACE, which is the default value set by INIT, to SET. For each word written to the bit map in SET mode, only bit map locations corresponding to logical '1' bits in the word are set to '1'. Bit map locations corresponding to logical '0' bits in the word remain unchanged. The various elements of the flight panel are then drawn using UBCMD, UCCMD, ULCMD, UFCMD, and STRCMD. As pointed out in the STRCMD description, STRCMD changes the drawing pattern. Therefore, following each series of STRCMD calls and before resuming drawing, AGCMD is called to reload the drawing pattern.

Procedures were then generated which, based on input from a joystick, update the various controls and indicators in the flight panel display. One of these procedures, ALTIMETER, updates the three hands on the altimeter indicator. (Refer to the listing of ALTIMETER in Appendix A.2.) First, an AGCMD call establishes solid line drawing, since the hands were drawn as solid lines. Next, a UMCMD call sets the drawing mode to CLEAR. For each word written to the bit map in CLEAR mode, only bit map locations corresponding to logical '1' bits in the word are set to '0'. Bit map locations corresponding to logical '0' bits in the word remain unchanged. The old hands are then redrawn using ULCMD, with the result that they are erased. A second UMCMD call changes the drawing mode to SET, after which the new hands are drawn using ULCMD. The rest of the flight panel display is updated in similar fashion.

Reactor Application

INTRODUCTION

This application example is a non-interactive simulation of a reactor (see Fig. 11). The fluids in the various closed systems comprising the reactor move in simulated flow. For this application, the iSBX 275 board is configured for color mode operation.

OVERVIEW

First, the necessary jumpers on the iSBX 275 board are detailed. Next, the process is described for determining the parameters for the INIT routine. Finally, the generation of the reactor display shown in Fig. 11 is discussed.

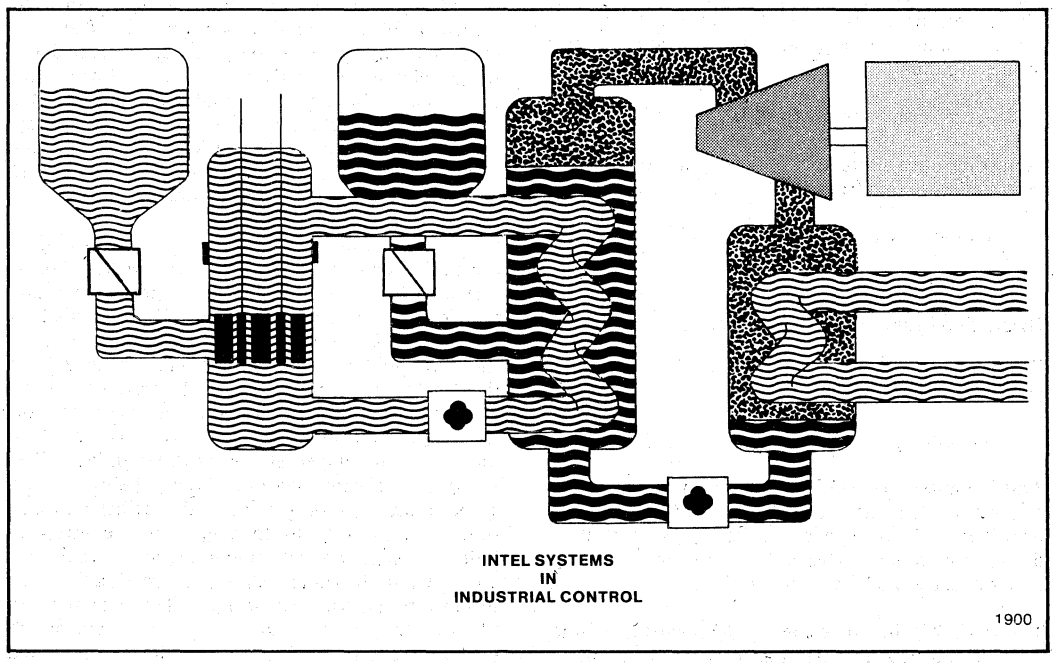


Figure 11. Reactor Display

JUMPERS

This application required the color capabilities of the iSBX 275 board. The board was therefore jumpered for color mode operation, resulting in a nominal resolution of 256 × 256. The board was also jumpered to provide an active low combined sync (CSYNC/) signal for the color monitor. All of the installed jumpers are described below.

1 - 2 test jumper
8 - 10 provides CSYNC/ output
12 - 13 selects color display mode
14 - 15 test jumper
17 - 18 selects divided-by-two mode

GRAPHICS INITIALIZATION

Introduction

In order to use the graphics library of PL/M-86 routines described earlier, the following include file was placed in the REACTOR module:

```
$include(/user/sbx275/ext/sbx275)
```

An application program must call the procedure INIT before it calls any other procedure from this graphics library. Typically, INIT is called in the main procedure of an application, as was done in REACTOR. (Refer to Fig. 12.) The iSBX 275 board is initialized according to the parameters accompanying the INIT call. These parameters specify the desired resolution, monitor, and mode of operation. They also specify the I/O port addresses of the iSBX 275 board. Following the INIT call, an application program may begin drawing by calling any of the other graphics procedures.

```
*************************************************
                    MAIN
*************************************************

  925   call init(4096,288,227,2,3,2,3,16,16,
                                  12h,82h,80h);
  926   call reactor;
  927   call dqexit(0);

  928   end reactor$module;
```

Figure 12. INIT Call In REACTOR

Overview

Many steps are involved in determining the parameters for INIT. First, the number and size of the bit map planes must be specified, providing the nominal resolution. Then the exact horizontal resolution and the associated monitor timing parameters must be determined. Next, the vertical resolution and the associated monitor timing parameters must be determined. Finally, the mode of operation and I/O addresses of the iSBX 275 board must be specified. The INIT parameters for REACTOR were determined as described below.

Color Plane Size

The reactor application requires the color display mode of the iSBX 275 board. Referring to the description of the plane_size parameter, we find for this mode that

plane_size = 4096 (or 1000h).

Horizontal Resolution

The reactor application requires the non-interlaced display mode of the monitor. Referring to the description of the xres parameter, we find that the horizontal timing parameters must satisfy Eq. 5x. Given the 12.6 MHz oscillator shipped on the iSBX 275 board, the divide-by-two mode, and the monitor horizontal rate of 15.75 KHz, Eq. 5x becomes

$$hs + hfp + hbp + AW = \frac{6.3\ \text{MHz}}{16 \times 15.75\ \text{KHz}} = 25, \qquad \textit{(Eq. P1)}$$

which is an integer value as required.

Still referring to the xres parameter description, we find that the maximum value for AW is determined using Eq. 7x. A 4:3 aspect ratio is desired, so given the nominal 255mm:190mm aspect ratio of the monitor, Eq. 7x becomes

$$AW \leqslant 16 \times \sqrt{255\text{mm} \div 90\text{mm}} = 18.54,$$

which, when rounded down as required to the nearest even integer, becomes

$$AW \leqslant 18.$$

The monitor manual specifies a minimum horizontal sync of 4.4 μs. Referring to the description of the hs parameter, we find that the minimum value for hs must be

$$hs \geqslant 4.4\ \mu s \times 6.3\ \text{MHz} \div 16 = 1.733,$$

which, when rounded up as required to the nearest integer, becomes

$$hs \geqslant 2.$$

The monitor manual specifies a minimum horizontal front porch of 1.2 μs. Referring to the description of the hfp parameter, we find that the minimum value for hfp must be the greater of '2' and

$$hfp \geq 1.2\ \mu s \times 6.3\ MHz \div 16 = 0.473,$$

and since $2 > 0.473$,

$$hfp \geq 2.$$

The monitor manual specifies a minimum horizontal back porch of 4.9 μs. Referring to the description of the hbp parameter, we find that the minimum value for hbp must be the greater of '3' and

$$hbp = 4.9\ \mu s \times 6.3\ MHz \div 16 = 1.929,$$

and since $3 > 1.929$,

$$hbp \geq 3.$$

Using the values determined above,

$$hs+hfp+hbp+AW=2+2+3+18=25,$$

satisfying Eq. P1 above. The horizontal parameter values are thus

$$AW = 18,\ hs = 2,\ hfp = 2,\ hbp = 3.$$

Referring once more to the description of the xres parameter, we find from Eq. 8x that

$$xres = 16 \times 18 = 288.$$

Vertical Resolution

Referring to the description of the yres parameter, we find that the vertical timing parameters must satisfy Eq. 5y. Given the monitor field rate of 60 Hz, Eq. 5y becomes

$$vs+vfp+vbp+AL=15.75\ KHz \div 60\ Hz=262.5,$$

which, when rounded down as required to the nearest integer, becomes

$$vs + vfp + vbp + AL = 262. \qquad \textit{(Eq. P2)}$$

Still referring to the yres parameter description, we find that the maximum value for AL is determined using Eq. 7y1, which becomes

$$AL \leq (256 \times 256) \div 288 = 227.6,$$

which, when rounded down as required to the nearest integer, becomes

$$AL \leq 227.$$

The monitor manual specifies a minimum vertical sync of three horizontal retrace-and-scan periods. Referring to the description of the vs parameter, we find that the minimum value for vs must be

$$vs \geq (3 \div 15.75\ KHz) \times 15.75\ KHz = 3.$$

The monitor manual specifies a minimum vertical front porch of three horizontal retrace-and-scan periods. Referring to the description of the vfp parameter, we find that the minimum value for vfp must be

$$vfp \geq (3 \div 15.75\ KHz) \times 15.75\ KHz = 3.$$

The monitor manual specifies a minimum vertical back porch of twelve horizontal retrace-and-scan periods. Referring to the description of the vbp parameter, we find that the minimum value for vbp must be

$$vbp \geq (12 \div 15.75\ KHz) \times 15.75\ KHz = 12.$$

Using the values determined above,

$$vs+vfp+vbp+AL=3+3+12+227=245,$$

which is seventeen less than required by Eq. P2 above. The vs, vfp, and vbp values determined above are minimums, while the AL value is a maximum. Since the sum of these values must be incremented by seventeen to satisfy Eq. P2, the vfp and vbp parameters are arbitrarily selected to be incremented by 13 and 4, respectively. The vertical parameter values are thus

$$AL = 227,\ vs = 3,\ vfp = 16,\ vbp = 16.$$

Referring once more to the description of the yres parameter, we find from Eq. 8y that

$$yres = 227.$$

Operation Mode and I/O Addresses

Referring to the description of the opmode parameter, we choose

$$opmode = 12H.$$

This initializes the GDC for non-interlaced display mode operation; for individual pixel addressing and drawing; and for drawing in display memory only during horizontal and vertical retrace, when display memory is blanked from the monitor.

The iSBX 275 board resides on SBX port J4 of the iSBC 86/30 board. Referring to the description of the command_port and data_port parameters, we find that the I/O address chosen for each of these

parameters must activate the MCS0/ signal on the SBX bus. Referring then to the iSBC 86/14 AND iSBC 86/30 SINGLE BOARD COMPUTER HARDWARE REFERENCE MANUAL, Intel order number 144044-002, page 3-7, we find that any of the I/O addresses 80, 82, 84, 86, 88, 8A, 8C, or 8E will activate MCS0/. Finally, referring to the iSBX 275 VIDEO GRAPHICS CONTROLLER MULTIMODULE BOARD REFERENCE MANUAL, Intel order number 144829-001, page 3-1, Table 3-1, we find that any of the I/O addresses 82, 86, 8A, or 8E may be used for the command_port parameter, and any of the I/O addresses 80, 84, 88, or 8C may be used for the data_port parameter. The I/O addresses chosen are therefore

command_port = 82H,

data_port = 80H.

CODE GENERATION

This section addresses the generation of the REACTOR bit map display. To begin with, before any code was generated, the reactor was drawn on grid paper. This paper was layed out according to the xres and yres parameter values determined above for the INIT routine. The paper had 288 squares in the horizontal direction and 227 squares in the vertical direction, so each square represented a pixel on the display. The lower left square was assigned the (x,y) coordinates (0,0), with x increasing to the right and y increasing up.

The code was then generated to draw the various reactor display elements in the bit map, according to their size and location on the grid paper. (Refer to Fig. 11.) This code calls UMCMD each time the color needs to be changed, and it draws the entire reactor display using ULCMD and UFCMD. It takes advantage of the fact that INIT clears display memory, initializes the drawing pattern for solid line drawing and area filling, and initializes the write mode to SET. For each word written to the bit map in SET mode, only bit map locations corresponding to logical '1' bits in the word are set to '1'. Bit map locations corresponding to logical '0' bits in the word remain unchanged.

REACTOR first sets the drawing color to WHITE. (Refer to the listing of REACTOR in Appendix B.) The outlines of the reactor vessel, the steam generator, and the condenser pipe are then drawn, and the turbine-to-generator shaft is filled in. Then the drawing color is changed to RED and the turbine is filled in. Next, the drawing color is changed to GREEN and the generator, the valve blocks, and the pump blocks are filled in. The drawing color is next changed to YELLOW. The valves and gears are then filled in over the green blocks, with the result that the valves and gears appear yellow. Then the drawing color is changed to MAGENTA, and the stationary reactor elements are filled in. Next, the drawing color is changed to CYAN and the moveable reactor elements are filled in. Finally, the drawing color is changed to BLUE. The fluids in the system are then filled in, followed by the steam in the steam generator.

Code was then generated to simulate motion in the fluids and steam. First, a UMCMD call changes the write mode from SET to COMPLEMENT. For each word written to the bit map in COMPLEMENT mode, only bit map locations corresponding to logical '1' bits in the word are complemented. Bit map locations corresponding to logical '0' bits in the word remain unchanged. For motion in the fluids, an AGCMD call establishes a dotted line drawing pattern. Every second line is then repeatedly rewritten, in sequence. This simulates motion by turning half of the pixels in these lines alternately off and on. Motion in the steam is simulated by repeatedly writing every second line twice, in sequence. First, an AGCMD call establishes the old line drawing pattern for a line. Then the line is redrawn, with the result that it is erased. A second AGCMD call establishes a new line drawing pattern for the line, and the new line is drawn.

PERFORMANCE CALCULATIONS

Introduction

The iSBX 275 board applications discussed in this application note have attempted to maximize display memory utilization and hence resolution. Since maximizing resolution may have the opposite effect on performance, this section discusses some of the performance and resolution tradeoffs which pertain to the iSBX 275 board.

For this discussion, performance refers to the speed with which an image can be updated in display memory. Typically, drawing in display memory is allowed only during retrace blanking, as in the examples described in this application note. The GDC can be initialized to allow drawing in display memory at any time (refer to the definition of the opmode parameter), but the resulting display flashing generally is not tolerable. Display flashing occurs when data is written to bit map locations which are being accessed for monitor refresh. Since a monitor screen is refreshed only during active display time, it is assumed here that the iSBX 275 board is always operated in flashless mode, such that drawing in display memory only occurs during retrace blanking.

The minimum bandwidth of the iSBX 275 board is 4 MHz. The maximum bandwidth is 10 MHz in color display mode and 13 MHz in black-and-white display

mode. In the divide-by-two mode, in which the oscillator frequency is divided by two, a 20 MHz oscillator may be used in color display mode, or a 25 MHz oscillator may be used in black-and-white display mode. The following performance discussion assumes that the 12.6 MHz oscillator shipped on the iSBX 275 board is used for all modes. It should be noted that for color display mode, performance can be nearly doubled by changing from the default oscillator (6.3 MHz bandwidth) to an oscillator which provides the full 10 MHz bandwidth available.

Overview

For the purpose of this discussion, a nominal 512 × 512 black-and-white image is the standard for comparison. Each pixel of this image is represented by one bit in display memory. The choice of color display mode limits the nominal resolution to 256 × 256 pixels. With one-fourth as many pixels, one might think that a color image would provide four times the performance of a 512 × 512 black-and-white image. However, each pixel of a color image is represented by three bits in display memory, one in each of three color planes. The primary colors are generated by drawing in one plane, the comlementary colors by drawing in two planes, and white is generated by drawing in all three planes. Since a color image typically contains a mixture of colors, it requires on the average that two bits be drawn in display memory for each pixel. This would imply that a color image provides twice the performance of a 512 × 512 black-and-white image. Since the default oscillator frequency must be divided in half in color display mode, though, there is essentially no difference in the drawing performance for a 256 × 256 8-color image and 512 × 512 black-and-white image.

If the default oscillator restriction is removed, a 256 × 256 8-color image can provide up to 60% better performance than a 512 × 512 black-and-white image. This is accomplished by replacing the default 12.6 MHz oscillator with a 20 MHz oscillator, with the iSBX 275 board in the divide-by-two mode; or by replacing the default oscillator with a 10 MHz oscillator, with the board in the non-divided mode. This increases the bandwidth of the board from 6.3 MHz to 10 MHz.

If a nominal resolution of 256 × 256 pixels is sufficient, a 256 × 256 black-and-white image can provide four times the performance of a 512 × 512 black-and-white image. This is due to the fact that the 512 × 512 image has four times as many pixels as the 256 × 256 image.

With drawing in display memory taking place only during retrace blanking, performance improvements can be realized by increasing the blanking time. Increasing the vertical blanking time decreases the number of display lines, and increasing the horizontal blanking time decreases the number of active words per line. Therefore, in order to maximize performance, the desired image should be drawn with the minimum allowable resolution. This minimizes the number of active display words comprising the image, thereby minimizing the time required to update the image in the bit map.

Available Drawing Time

The following equation expresses the fraction of each raster-scan cycle which is available for drawing in display memory. This fraction represents the theoretical limit on the drawing time available, enabling the calculation of the maximum theoretical performance of the iSBX 275 board. The larger the fraction, the higher the theoretical maximum performance, with a fully blanked display (which obviously has no intrinsic value) yielding a value of one. The actual performance is dependent on external factors, such as the amount of processing time which the CPU requires in order to generate the data for updating an image. For the equation below, the horizontal blanking time (hs + hfp + hbp) must be rounded down to the nearest even number. It is expressed in display words, as two display word times are necessary to draw each bit in display memory.

$$\text{FRACTION} = \frac{\left(\frac{hs+hfp+hbp}{hs+hfp+hbp+AW} \times AL\right) + (vs+vfp+vbp)}{vs+vfp+vbp+AL}$$

where

AW = number of active words,
AL = number of active lines,
hs + hfp + hbp = horizontal blanking time,
hs + hfp + hbp + AW = total horizontal time,
vs + vfp + vbp = vertical blanking time,
vs + vfp + vbp + AL = total vertical time.

As a general rule of thumb, taking into account external factors, the actual performance of the iSBX 275 board is approximated using

FRACTION = .25

The equation below expresses the performance of the iSBX 275 board in pixels per second, assuming graphics mode operation. Thirty-two oscillator cycles are consumed for each bit drawn in display memory.

$$\text{PERFORMANCE} = \frac{\text{FRACTION} \times (\text{iSBX 275 board bandwidth})}{32 \times (\text{number of color planes used})}$$

Plane Example

For the PLANE application discussed previously, the fraction of each raster-scan cycle available for drawing in display memory is

$$\text{FRACTION} = \frac{\left(\frac{5+4+5}{5+4+5+36} \times 226\right) + (4+16+16)}{4+16+16+226}$$

$$= \frac{(14 \times 226 \div 50) + 36}{262} = .379$$

The theoretical maximum performance of the iSBX 275 board for PLANE is therefore

$$\text{PERFORMANCE} = .379 \times (12.6\,\text{MHz}) \div 32$$

$$= 149{,}200 \text{ pixels per second,}$$

and the actual performance is approximated by

$$\text{PERFORMANCE} = .25 \times (12.6\,\text{MHz}) \div 32$$

$$= 98{,}400 \text{ pixels per second.}$$

Reactor Example

For the REACTOR application discussed previously, the fraction of each raster-scan cycle available for drawing in display memory is

$$\text{FRACTION} = \frac{\left(\frac{2+2+3}{2+2+3+18} \times 227\right) + (3+16+16)}{3+16+16+227}$$

$$= \frac{(7 \times 227 \div 25) + 35}{262} = .376$$

Assuming that all three color planes were written to for each color, which would happen only if everything were drawn in white, the theoretical maximum performance of the iSBX 275 board for REACTOR would be

$$\text{PERFORMANCE} = \frac{.376 \times (12.6\,\text{MHz} \div 2)}{32 \times 3}$$

$$= 24{,}600 \text{ pixels per second.}$$

Assuming that only one color plane were written to for each color, which would happen only if everything were drawn in primary colors, the theoretical maximum performance of the iSBX 275 board for REACTOR would be

$$\text{PERFORMANCE} = \frac{.376 \times (12.6\,\text{MHz} \div 2)}{32}$$

$$= 74{,}000 \text{ pixels per second.}$$

Since all seven available colors are used, each color drawn in display memory requires writing to an average of two color planes. The theoretical maximum performance of the iSBX 275 board for REACTOR is therefore best represented by

$$\text{PERFORMANCE} = \frac{.376 \times (12.6\,\text{MHz} \div 2)}{32 \times 2}$$

$$= 37{,}000 \text{ pixels per second,}$$

and the actual performance is approximated by

$$\text{PERFORMANCE} = \frac{.25 \times (12.6\,\text{MHz} \div 2)}{32 \times 2}$$

$$= 24{,}600 \text{ pixels per second.}$$

Image Updating

From the performance numbers derived above, relative display image update times can be calculated. The actual number of pixels involved in an image update is application dependent. Therefore, for this discussion, it will be assumed that a typical image update involves half of the pixels displayed. The following equation represents the time required to update a display image.

$$\text{UPDATE} = \frac{\text{number of pixels updated}}{\text{PERFORMANCE}}$$

For a black-and-white display image, the maximum possible resolution is

$$\text{RESOLUTION} = 512 \times 512 = 262{,}144 \text{ pixels.}$$

Therefore, using (RESOLUTION ÷ 2) as the number of pixels updated, the time required to update a 512 × 512 resolution black-and-white display image is approximated by

$$\text{UPDATE} = \frac{262{,}144 \div 2}{98{,}400} = 1.332 \text{ seconds.}$$

For a color display image, the maximum possible resolution is

$$\text{RESOLUTION} = 256 \times 256 = 65{,}536 \text{ pixels.}$$

Therefore, using (RESOLUTION÷2) as the number of pixels updated, the time required to update a 256 × 256 resolution 8-color display image is approximated by

$$\text{UPDATE} = \frac{65{,}536 \div 2}{24{,}600} = 1.332 \text{ seconds.}$$

Performance Summary

The iSBX 275 board is capable of drawing at a rate of 100,000 pixels per second in black-and-white mode. This assumes a bandwidth of 12.6 MHz. By changing the oscillator to allow for a 10 MHz bandwidth, the iSBX 275 board is capable of drawing at a rate of 40,000 pixels per second in color mode.

CONCLUSIONS

The iSBX 275 board provides an easy-to-use, configurable graphics capability for boards and systems which support the SBX bus. Color, resolution, and performance are design criteria which can be weighed against each other.

With the addition of the graphics routines described in this application note, the iSBX 275 board reduces graphical image generation to little more than a series of PL/M-86 calls. The two application examples verified that display images of varying complexity can be drawn using a handful of simple figure drawing commands. Circles, arcs, lines, rectangles, characters, character strings, and filled rectangular areas are each drawn by calling a single command. In addition, a single command is all that is required to change the drawing color, line style, or fill pattern, or to define a random 8 × 8 symbol.

Not only is the iSBX 275 board easy to use, but the general board setup and initialization sequence described in this application note will be required by only a handful of users. Most users will find that they can simply use the initializaiton parameters calculated in the two application examples, without going through the process of calculating the parameters for themselves.

APPENDIX A

PLANE DRAW$PANEL listing

```
                $eject

                /*****************************************************************
                                        Draw Panel

                *****************************************************************/
820     1       draw$panel: procedure ;
821     2           declare n byte;

                    /* clear memory and display */
822     2           call upcmd (0, 0, bw_plane);
823     2           call awcmd (replace,east,zeros, 4000h);

                    /* set pattern for full line style */
824     2           call agcmd (0ffffh, 0ffffh, 0ffffh, 0ffffh);

                    /* draw com control */
825     2           call umcmd (bw, set);
826     2           call ubcmd (088,112,65,35,east);
827     2           call ubcmd (095,119,51,11,east);
828     2           call strcmd (104,136,normal_char,east,one_x,bw,4,@com1$label);
829     2           call strcmd (120,120,normal_char,east,one_x,bw,1,@decimal);

830     2           do n = 0 to 4;
831     3               call strcmd (com$locn(n),120,normal_char,east,one_x,bw,
                                1,@digits(com$freq(n)));
832     3           end;

                    /* draw nav 1 control */
833     2           call agcmd (0ffffh,0ffffh,0ffffh,0ffffh);
834     2           call ubcmd (088,072,65,35,east);
835     2           call ubcmd (095,079,51,11,east);
836     2           call strcmd (104,096,normal_char,east,one_x,bw,4,@nav1$label);
837     2           call strcmd (120,080,normal_char,east,one_x,bw,1,@decimal);

838     2           do n = 0 to 4;
839     3               call strcmd (nav1$locn(n),80,normal_char,east,one_x,bw,
                                1,@digits(nav1$freq(n)));
840     3           end;

                    /* draw nav 2 control */
841     2           call agcmd (0ffffh,0ffffh,0ffffh,0ffffh);
842     2           call ubcmd (088,032,65,35,east);
843     2           call ubcmd (095,039,51,11,east);
844     2           call strcmd (104,056,normal_char,east,one_x,bw,4,@nav2$label);
845     2           call strcmd (120,040,normal_char,east,one_x,bw,1,@decimal);

846     2           do n = 0 to 4;
847     3               call strcmd (nav2$locn(n),40,normal_char,east,one_x,bw,
                                1,@digits(nav2$freq(n)));
848     3           end;

                    /* draw adf control */
849     2           call agcmd (0ffffh,0ffffh,0ffffh,0ffffh);
850     2           call ubcmd (168,112,65,35,east);
851     2           call ubcmd (183,119,35,11,east);
852     2           call strcmd (188,136,normal_char,east,one_x,bw,3,@adf$label);
853     2           do n = 0 to 3;
854     3               call strcmd (adf$locn(n),120,normal_char,east,one_x,bw,
                                1,@digits(adf$freq(n)));
855     3           end;
```

```
                /* draw transponder control */
856   2         call agcmd (0ffffh,0ffffh,0ffffh,0ffffh);
857   2         call ubcmd (168,072,65,35,east);
858   2         call ubcmd (183,079,35,11,east);
859   2         call strcmd (180,096,normal_char,east,one_x,bw,
                        5,@trans$label);

860   2         do n = 0 to 3;
861   3             call strcmd (tran$locn(n),80,normal_char,east,one_x,bw,
                            1,@digits(tran$freq(n)));
862   3         end;

                /* draw key board control */
863   2         call agcmd (0ffffh,0ffffh,0ffffh,0ffffh);
864   2         call ubcmd (168,032,65,35,east);
865   2         call ubcmd (175,039,51,11,east);
866   2         call strcmd (176,056,normal_char,east,one_x,bw,
                        6,@keybrd$label);
867   2         call strcmd (200,40,normal_char,east,one_x,bw,1,@decimal);

                /* draw marker beacon */
868   2         call agcmd (0ffffh,0ffffh,0ffffh,0ffffh);
869   2         call ubcmd (248,380,81,45,east);
870   2         call ubcmd (256,384,17,17,east);
871   2         call ubcmd (280,384,17,17,east);
872   2         call ubcmd (304,384,17,17,east);
873   2         call strcmd (260,408,normal_char,east,one_x,bw,1,@o$label);
874   2         call strcmd (284,408,normal_char,east,one_x,bw,1,@m$label);
875   2         call strcmd (308,408,normal_char,east,one_x,bw,1,@i$label);

                /* draw timer */
876   2         call agcmd (0ffffh,0ffffh,0ffffh,0ffffh);
877   2         call ubcmd (439,391,43,11,east);
878   2         call ubcmd (432,384,57,35,east);
879   2         call strcmd (440,408,normal_char,east,one_x,bw,
                        5,@timer$label);
880   2         call strcmd (456,392,normal_char,east,one_x,bw,1,@colon);

                /* draw circle for airspeed indicator */
881   2         call agcmd (0ffffh, 0ffffh, 0ffffh, 0ffffh);
882   2         call uccmd (073h, 144h, 34h);

883   2         call ulcmd (116,376,116,374);
884   2         call ulcmd (151,359,153,361);
885   2         call ulcmd (157,354,158,355);
886   2         call ulcmd (161,347,162,348);
887   2         call ulcmd (165,340,165,340);
888   2         call ulcmd (165,332,167,332);
889   2         call ulcmd (167,324,168,324);
890   2         call ulcmd (165,316,167,316);
891   2         call ulcmd (165,308,165,308);
892   2         call ulcmd (161,301,162,300);
893   2         call ulcmd (157,294,158,293);
894   2         call ulcmd (151,289,153,287);
895   2         call ulcmd (146,283,147,282);
896   2         call ulcmd (139,279,140,278);
897   2         call ulcmd (132,275,132,275);
898   2         call ulcmd (124,275,124,273);
899   2         call ulcmd (116,273,116,272);
900   2         call ulcmd (108,275,108,273);
901   2         call ulcmd (100,275,100,275);
902   2         call ulcmd ( 93,279, 92,278);
903   2         call ulcmd ( 86,283, 85,282);
904   2         call ulcmd ( 81,289, 79,287);
905   2         call ulcmd ( 75,294, 74,293);
906   2         call ulcmd ( 71,301, 70,300);
```

```
907   2          call ulcmd ( 67,308, 67,308);
908   2          call ulcmd ( 67,316, 65,316);
909   2          call ulcmd ( 65,324, 64,324);
910   2          call ulcmd ( 67,332, 65,332);

911   2          call strcmd (112,364,normal_char,east,one_x,bw,1,@zero);
912   2          call strcmd (136,348,normal_char,east,one_x,bw,2,@fifty);
913   2          call strcmd (148,320,normal_char,east,one_x,bw,
                            2,@seventyfive);
914   2          call strcmd (132,292,normal_char,east,one_x,bw,3,@hundred);
915   2          call strcmd (104,276,normal_char,east,one_x,bw,
                            3,@onetwentyfive);
916   2          call strcmd ( 76,292,normal_char,east,one_x,bw,3,@onefifty);
917   2          call strcmd ( 68,320,normal_char,east,one_x,bw,
                            3,@oneseventyfive);

                /* draw turn and bank indicator */
918   2          call agcmd (0ffffh, 0ffffh, 0ffffh, 0ffffh);
919   2          call uccmd (073h, 0d8h, 34h);
920   2          call ubcmd (05ah, 0b7h, 51, 18, east);
921   2          call ufcmd (06fh,0f8h,8,9,east);
922   2          call ulcmd (063h,0f4h,06ah,0f6h);
923   2          call ulcmd (06ah,0f6h,070h,0feh);
924   2          call ulcmd (070h,0feh,060h,0fch);
925   2          call ulcmd (060h,0fch,063h,0f4h);
926   2          call ulcmd (083h,0f4h,086h,0fch);
927   2          call ulcmd (086h,0fch,07fh,0feh);
928   2          call ulcmd (07fh,0feh,07dh,0f6h);
929   2          call ulcmd (07dh,0f6h,083h,0f4h);

930   2          do n = 0 to 7;
931   3              call ulcmd(tabx1(n),taby1(n),tabx2(n),taby2(n));
932   3          end;

933   2          call ufcmd (06fh,0b8h,8,9,east);

                /* draw artificial horizen indicator */
934   2          call uccmd (0e6h, 144h, 34h);

                /* draw directional gyro */
935   2          call uccmd (0e6h, 0d8h, 34h);
936   2          call ubcmd (0c7h, 0d4h, 63, 17, east);

937   2          call ulcmd (0e6h,0d8h,0e6h,0d6h);

                /* draw altimeter */
938   2          call uccmd (159h, 144h, 34h);
939   2          call strcmd (155h,170h,normal_char,east,one_x,bw,
                             length(zero), @zero);
940   2          call strcmd (171h,166h,normal_char,east,one_x,bw,
                             length(one), @one);
941   2          call strcmd (183h,150h,normal_char,east,one_x,bw,
                             length(two), @two);
942   2          call strcmd (183h,130h,normal_char,east,one_x,bw,
                             length(three), @three);
943   2          call strcmd (171h,118h,normal_char,east,one_x,bw,
                             length(four), @four);
944   2          call strcmd (155h,111h,normal_char,east,one_x,bw,
                             length(five), @five);
945   2          call strcmd (139h,118h,normal_char,east,one_x,bw,
                             length(six), @six);
946   2          call strcmd (12bh,130h,normal_char,east,one_x,bw,
                             length(seven), @seven);
947   2          call strcmd (12bh,150h,normal_char,east,one_x,bw,
                             length(eight), @eight);
948   2          call strcmd (139h,166h,normal_char,east,one_x,bw,
                             length(nine), @nine);
```

```
                /* draw rate of climb */
949     2       call agcmd (0ffffh, 0ffffh, 0ffffh, 0ffffh);
950     2       call uccmd (159h, 0d8h, 34h);

951     2       call strcmd (155h,0fch,normal_char,east,one_x,bw,
                             length(one), @one);
952     2       call strcmd (167h,0f4h,normal_char,east,one_x,bw,
                             length(one$point$five), @one$point$five);
953     2       call strcmd (17eh,0d4h,normal_char,east,one_x,bw,
                             length(two), @two);
954     2       call strcmd (167h,0b8h,normal_char,east,one_x,bw,
                             length(one$point$five), @one$point$five);
955     2       call strcmd (155h,0adh,normal_char,east,one_x,bw,
                             length(one), @one);
956     2       call strcmd (131h,0f4h,normal_char,east,one_x,bw,
                             length(half), @half);
957     2       call strcmd (131h,0b8h,normal_char,east,one_x,bw,
                             length(half), @half);
958     2       call strcmd (12ch,0d4h,normal_char,east,one_x,bw,
                             length(zero), @zero);

                /* draw tachometer */
959     2       call agcmd(0ffffh, 0ffffh, 0ffffh, 0ffffh);
960     2       call uccmd (159h, 06ch, 34h);
961     2       call ulcmd (308,71,309,72);
962     2       call ulcmd (303,77,304,78);
963     2       call ulcmd (298,84,300,85);
964     2       call ulcmd (295,92,297,92);
965     2       call ulcmd (293,100,295,100);
966     2       call ulcmd (293,108,295,108);
967     2       call ulcmd (293,116,295,116);
968     2       call ulcmd (295,124,297,123);
969     2       call ulcmd (298,132,300,131);
970     2       call ulcmd (303,139,305,137);
971     2       call ulcmd (308,145,310,143);
972     2       call ulcmd (314,150,316,148);
973     2       call ulcmd (321,154,322,153);
974     2       call ulcmd (329,157,330,156);
975     2       call ulcmd (337,159,337,157);
976     2       call ulcmd (345,160,345,158);
977     2       call ulcmd (353,159,353,157);
978     2       call ulcmd (361,157,360,156);
979     2       call ulcmd (369,154,368,153);
980     2       call ulcmd (376,150,374,148);
981     2       call ulcmd (382,145,380,143);
982     2       call ulcmd (387,139,385,137);
983     2       call ulcmd (391,132,390,131);
984     2       call ulcmd (394,124,393,123);
985     2       call ulcmd (396,116,394,116);
986     2       call ulcmd (397,108,395,108);
987     2       call ulcmd (396,100,394,100);
988     2       call ulcmd (394,92,393,93);
989     2       call ulcmd (391,84,390,85);
990     2       call ulcmd (387,77,385,79);
991     2       call ulcmd (382,71,380,73);
992     2       call strcmd (309,72,normal_char,east,one_x,bw,1,@zero);
993     2       call strcmd (297,104,normal_char,east,one_x,bw,1,@five);
994     2       call strcmd (309,136,normal_char,east,one_x,bw,2,@ten);
995     2       call strcmd (337,148,normal_char,east,one_x,bw,2,@fifteen);
996     2       call strcmd (365,136,normal_char,east,one_x,bw,2,@twenty);
997     2       call strcmd (377,104,normal_char,east,one_x,bw,2,@twentyfive);
998     2       call strcmd (365,72,normal_char,east,one_x,bw,2,@thirty);
999     2       call strcmd (333,80,normal_char,east,one_x,bw,3,@rpm$label);

                /* draw vor #1 indicator */
1000    2       call agcmd (0ffffh, 0ffffh, 0ffffh, 0ffffh);
```

```
1001   2          call uccmd (1cch, 144h, 34h);
1002   2          call ulcmd (456,327,456,321);
1003   2          call ulcmd (457,328,463,328);
1004   2          call ulcmd (464,327,464,321);
1005   2          call ulcmd (457,320,463,320);
1006   2          call strcmd (428,304,normal_char,east,one_x,bw,3,@obs$label);
1007   2          call strcmd (468,348,normal_char,east,one_x,bw,3,@dst$label);
1008   2          call strcmd (428,348,normal_char,east,one_x,bw,3,@nav$label);
1009   2          call strcmd (436,340,normal_char,east,one_x,bw,1,@one);

                   /* draw vor #2 indicator */
1010   2          call agcmd (0fffffh, 0fffffh, 0fffffh, 0fffffh);
1011   2          call uccmd (1cch, 0d8h, 34h);
1012   2          call ulcmd (456,219,456,213);
1013   2          call ulcmd (457,220,463,220);
1014   2          call ulcmd (464,219,464,213);
1015   2          call ulcmd (457,212,463,212);
1016   2          call strcmd (428,196,normal_char,east,one_x,bw,3,@obs$label);
1017   2          call strcmd (428,240,normal_char,east,one_x,bw,3,@nav$label);
1018   2          call strcmd (436,232,normal_char,east,one_x,bw,1,@two);

                   /* draw adf indicator */
1019   2          call agcmd (0fffffh,0fffffh,0fffffh,0fffffh);
1020   2          call uccmd (1cch, 06ch, 34h);
1021   2          call ulcmd (460,160,460,158);
1022   2          call ulcmd (469,159,468,157);
1023   2          call ulcmd (477,156,477,154);
1024   2          call ulcmd (485,153,484,151);
1025   2          call ulcmd (493,147,492,146);
1026   2          call ulcmd (499,141,498,140);
1027   2          call ulcmd (505,134,503,133);
1028   2          call ulcmd (508,125,506,125);
1029   2          call ulcmd (511,117,509,116);
1030   2          call ulcmd (512,108,510,108);
1031   2          call ulcmd (511, 98,509, 99);
1032   2          call ulcmd (508, 90,506, 90);
1033   2          call ulcmd (505, 82,503, 83);
1034   2          call ulcmd (499, 74,498, 75);
1035   2          call ulcmd (493, 68,492, 69);
1036   2          call ulcmd (486, 62,485, 64);
1037   2          call ulcmd (477, 59,477, 61);
1038   2          call ulcmd (469, 56,468, 58);
1039   2          call ulcmd (460, 56,460, 58);
1040   2          call ulcmd (450, 56,451, 58);
1041   2          call ulcmd (442, 59,442, 61);
1042   2          call ulcmd (434, 62,435, 64);
1043   2          call ulcmd (426, 68,427, 69);
1044   2          call ulcmd (420, 74,421, 75);
1045   2          call ulcmd (414, 81,416, 82);
1046   2          call ulcmd (411, 90,413, 90);
1047   2          call ulcmd (408, 98,410, 99);
1048   2          call ulcmd (408,108,410,108);
1049   2          call ulcmd (408,117,410,116);
1050   2          call ulcmd (411,125,413,125);
1051   2          call ulcmd (414,133,416,132);
1052   2          call ulcmd (420,141,421,140);
1053   2          call ulcmd (426,147,427,146);
1054   2          call ulcmd (433,153,434,151);
1055   2          call ulcmd (442,156,442,154);
1056   2          call ulcmd (450,159,451,157);
1057   2          call strcmd (456,148,normal_char,east,one_x,bw,1,@zero);
1058   2          call strcmd (478,142,normal_char,east,one_x,bw,1,@three);
1059   2          call strcmd (494,126,normal_char,east,one_x,bw,1,@six);
1060   2          call strcmd (500,104,normal_char,east,one_x,bw,1,@nine);
1061   2          call strcmd (488, 88,normal_char,east,one_x,bw,2,@twelve);
1062   2          call strcmd (476, 68,normal_char,east,one_x,bw,2,@fifteen);
```

```
1063   2            call strcmd (452,60,normal_char,east,one_x,bw,2,@eighteen);
1064   2            call strcmd (429,68,normal_char,east,one_x,bw,2,@twentyone);
1065   2            call strcmd (416,84,normal_char,east,one_x,bw,2,@twentyfour);
1066   2            call strcmd (412,104,normal_char,east,one_x,bw,
                                 2,@twentyseven);
1067   2            call strcmd (416,126,normal_char,east,one_x,bw,2,@thirty);
1068   2            call strcmd (429,142,normal_char,east,one_x,bw,
                                 2,@thirtythree);
1069   2            call strcmd (448,80,normal_char,east,one_x,bw,3,@adf$label);


1070   2            call agcmd (0ffffh, 0ffffh, 0ffffh, 0ffffh);

1071   2            call umcmd(bw,compliment);
1072   2       end draw$panel;
```

PLANE ALTIMETER listing

```
                $eject

                /*****************************************************************

                This is the altimeter display program. It is called using the
                actual altitude as a parameter.

                *****************************************************************/
615   1         altimeter: procedure (altitude) ;

616   2             declare altitude real;
617   2             declare (x21,y21,x22,y22,x23,y23) word;

618   2             declare x$org literally '159h';
619   2             declare y$org literally '144h';

                    /* save old values */
620   2             x21 = x$alt$10k;
621   2             x22 = x$alt$1k;
622   2             x23 = x$alt$100;
623   2             y21 = y$alt$10k;
624   2             y22 = y$alt$1k;
625   2             y23 = y$alt$100;

                    /* draw ten thousand foot pointer */
626   2             x$alt$10k = unsign(fix(16. * mqerCOS((90. - ((altitude/10000.)
                          * 36.)) * deg_to_rad))) + x$org;
627   2             y$alt$10k = unsign(fix(16. * mqerSIN((90. - ((altitude/10000.)
                          * 36.)) * deg_to_rad))) + y$org;
628   2             if (altitude > 10000.) then altitude = altitude - 10000.;

                    /* draw one thousand foot pointer */
630   2             x$alt$1k = unsign(fix(28. * mqerCOS((90. - ((altitude/1000.)
                         * 36.)) * deg_to_rad))) + x$org;
631   2             y$alt$1k = unsign(fix(28. * mqerSIN((90. - ((altitude/1000.)
                         * 36.)) * deg_to_rad))) + y$org;
632   2             do while (altitude > 1000.);
633   3                 altitude = altitude - 1000.;
634   3                 end;

                    /* draw one hundred foot pointer */
635   2             x$alt$100 = unsign(fix(40. * mqerCOS((90. - ((altitude/100.)
                          * 36.)) * deg_to_rad))) + x$org;
636   2             y$alt$100 = unsign(fix(40. * mqerSIN((90. - ((altitude/100.)
                          * 36.)) * deg_to_rad))) + y$org;

                    /* erase old hands */
637   2             call agcmd (0fffffh, 0fffffh, 0fffffh, 0fffffh);
638   2             call umcmd (bw, clear);
639   2             call ulcmd (x$org,y$org,x21,y21);
640   2             call ulcmd (x$org,y$org,x22,y22);
641   2             call ulcmd (x$org,y$org,x23,y23);

                    /* draw new hands */
642   2             call umcmd (bw, set);
643   2             call ulcmd (x$org,y$org,x$alt$10k,y$alt$10k);
644   2             call ulcmd (x$org,y$org,x$alt$1k,y$alt$1k);
645   2             call ulcmd (x$org,y$org,x$alt$100,y$alt$100);

                    /* return */
646   2             return;

647   2         end altimeter;
```

APPENDIX B

REACTOR listing

```
iRMX 86 PL/M-86 V2.3 COMPILATION OF MODULE REACTORMODULE
OBJECT MODULE PLACED IN /USER/SBX275/OBJ/REACTOR
COMPILER INVOKED BY:  :LANG:plm86 /USER/SBX275/SRC/REACTOR PRINT(/USER/SBX275/LS
T/REACTOR)
                      OBJECT(/USER/SBX275/OBJ/REACTOR) LARGE


   1          reactor$module: do;
              $nolist

  95    1       reactor: procedure reentrant public;

  96    2       declare   cycle word,
                          (i,j,k,l) byte,
                          (x1,y1,x2,y2) word;

  97    2       declare pattern (4) word data (4224h,
                                                 8a92h,
                                                 2429h,
                                                 484ah);

  98    2       declare line1 (*) byte data ('286 SYSTEMS'),
                        line2 (*) byte data ('in'),
                        line3 (*) byte data ('INDUSTRIAL CONTROL');

                  /********* DRAW OUTLINE *********/

                  /* set color and write mode */
  99    2         call umcmd(white,set);

                  /* draw reactor vessel outline */
 100    2         call ulcmd(23,152,23,158);
 101    2         call ulcmd(23,158,9,172);
 102    2         call ulcmd(9,172,9,203);
 103    2         call ulcmd(9,203,11,205);
 104    2         call ulcmd(11,205,44,205);
 105    2         call ulcmd(44,205,46,203);
 106    2         call ulcmd(46,203,46,172);
 107    2         call ulcmd(46,172,32,158);
 108    2         call ulcmd(32,158,32,152);

 109    2         call ulcmd(32,141,32,136);
 110    2         call ulcmd(33,135,50,135);
 111    2         call ulcmd(50,135,50,173);
 112    2         call ulcmd(50,173,52,175);
 113    2         call ulcmd(52,175,78,175);
 114    2         call ulcmd(78,175,80,173);
 115    2         call ulcmd(80,173,80,165);
 116    2         call ulcmd(80,165,151,165);
 117    2         call ulcmd(152,164,153,164);
 118    2         call ulcmd(154,163,155,163);
 119    2         call ulcmd(155,163,155,162);
 120    2         call ulcmd(156,161,156,158);
 121    2         call ulcmd(155,157,155,156);
 122    2         call ulcmd(155,156,147,148);

 123    2         call ulcmd(144,150,156,138);
 124    2         call ulcmd(156,137,157,136);
 125    2         call ulcmd(157,135,156,134);
 126    2         call ulcmd(156,133,147,124);
```

```
127   2          call ulcmd(144,126,155,115);
128   2          call ulcmd(155,115,155,114);
129   2          call ulcmd(156,113,156,110);
130   2          call ulcmd(155,109,155,108);
131   2          call ulcmd(155,108,154,108);
132   2          call ulcmd(153,107,152,107);
133   2          call ulcmd(151,106,125,106);

134   2          call ulcmd(23,141,23,128);
135   2          call ulcmd(23,128,25,126);
136   2          call ulcmd(25,126,50,126);
137   2          call ulcmd(50,126,50,101);
138   2          call ulcmd(50,101,52,99);
139   2          call ulcmd(52,99,78,99);
140   2          call ulcmd(78,99,80,101);
141   2          call ulcmd(80,101,80,106);
142   2          call ulcmd(80,106,108,106);

143   2          call ulcmd(108,115,80,115);
144   2          call ulcmd(80,115,80,156);
145   2          call ulcmd(80,156,146,156);

146   2          call ulcmd(140,155,135,150);
147   2          call ulcmd(135,149,134,148);
148   2          call ulcmd(134,147,135,146);
149   2          call ulcmd(135,145,147,133);

150   2          call ulcmd(144,135,135,126);
151   2          call ulcmd(135,125,134,124);
152   2          call ulcmd(134,123,135,122);
153   2          call ulcmd(135,121,144,112);

154   2          call ulcmd(141,115,125,115);

                /* draw boxes on reactor vessel sides */
155   2          call ufcmd(47,146,3,8,east);
156   2          call ufcmd(81,146,3,8,east);

                /* draw steam generator outline */
157   2          call ulcmd(100,152,100,155);
158   2          call ulcmd(92,166,86,172);
159   2          call ulcmd(86,172,86,203);
160   2          call ulcmd(86,203,88,205);
161   2          call ulcmd(88,205,121,205);
162   2          call ulcmd(121,205,123,203);
163   2          call ulcmd(123,203,123,172);
164   2          call ulcmd(123,172,117,166);
165   2          call ulcmd(109,155,109,152);

166   2          call ulcmd(109,141,109,136);
167   2          call ulcmd(110,135,129,135);
168   2          call ulcmd(129,135,129,155);
169   2          call ulcmd(129,166,129,188);
170   2          call ulcmd(129,188,131,190);
171   2          call ulcmd(131,190,141,190);
172   2          call ulcmd(141,190,141,203);
173   2          call ulcmd(141,203,143,205);
174   2          call ulcmd(143,205,195,205);
175   2          call ulcmd(195,205,197,203);
176   2          call ulcmd(197,203,197,194);

177   2          call ulcmd(100,141,100,128);
178   2          call ulcmd(100,128,102,126);
179   2          call ulcmd(102,126,129,126);
180   2          call ulcmd(129,126,129,116);
181   2          call ulcmd(129,105,129,101);
```

```
182   2           call ulcmd(129,101,131,99);
183   2           call ulcmd(131,99,141,99);
184   2           call ulcmd(141,99,141,82);
185   2           call ulcmd(141,82,143,80);
186   2           call ulcmd(143,80,172,80);

187   2           call ulcmd(172,89,151,89);
188   2           call ulcmd(150,90,150,99);
189   2           call ulcmd(150,99,160,99);
190   2           call ulcmd(160,99,162,101);
191   2           call ulcmd(162,101,162,188);
192   2           call ulcmd(162,188,160,190);
193   2           call ulcmd(160,190,150,190);
194   2           call ulcmd(150,190,150,195);
195   2           call ulcmd(151,196,187,196);
196   2           call ulcmd(188,195,188,191);

197   2           call ulcmd(203,167,203,159);
198   2           call ulcmd(203,159,193,159);
199   2           call ulcmd(193,159,191,157);
200   2           call ulcmd(191,157,191,101);
201   2           call ulcmd(191,101,193,99);
202   2           call ulcmd(193,99,203,99);
203   2           call ulcmd(203,99,203,90);
204   2           call ulcmd(202,89,189,89);

205   2           call ulcmd(189,80,210,80);
206   2           call ulcmd(210,80,212,82);
207   2           call ulcmd(212,82,212,99);
208   2           call ulcmd(212,99,222,99);
209   2           call ulcmd(222,99,224,101);
210   2           call ulcmd(224,101,224,115);
211   2           call ulcmd(224,126,224,141);
212   2           call ulcmd(224,152,224,157);
213   2           call ulcmd(224,157,222,159);
214   2           call ulcmd(222,159,212,159);
215   2           call ulcmd(212,159,212,164);

                  /* draw condenser pipe */
216   2           call ulcmd(287,151,202,151);
217   2           call ulcmd(201,150,200,150);
218   2           call ulcmd(199,149,198,149);
219   2           call ulcmd(198,149,198,148);
220   2           call ulcmd(197,147,197,144);
221   2           call ulcmd(198,143,198,142);
222   2           call ulcmd(198,142,206,134);

223   2           call ulcmd(209,136,198,125);
224   2           call ulcmd(198,125,198,124);
225   2           call ulcmd(197,123,197,120);
226   2           call ulcmd(198,119,198,118);
227   2           call ulcmd(198,118,199,118);
228   2           call ulcmd(200,117,201,117);
229   2           call ulcmd(202,116,287,116);

230   2           call ulcmd(287,125,212,125);

231   2           call ulcmd(209,122,218,131);
232   2           call ulcmd(218,132,219,133);
233   2           call ulcmd(219,134,218,135);
234   2           call ulcmd(218,136,213,141);

235   2           call ulcmd(207,142,287,142);

                  /********* DRAW OBJECTS *********/
```

```
                /* draw turbine-to-generator shaft */
236   2         call ufcmd(220,179,10,6,east);

                /* change color */
237   2         call umcmd(red,set);

                /* draw turbine */
238   2         x1 = 181;
239   2         y1 = 175;
240   2         y2 = 14;

241   2         do while x1 < 220;
242   3           call ufcmd(x1,y1,3,y2,east);
243   3           x1 = x1 + 3;
244   3           y1 = y1 - 1;
245   3           y2 = y2 + 2;
246   3         end;

                /* change color */
247   2         call umcmd(green,set);

                /* draw generator */
248   2         call ufcmd(230,166,48,32,east);
249   2         call ufcmd(231,165,46,34,east);
250   2         call ufcmd(232,164,44,36,east);

                /* draw valve and pump blocks */
251   2         call ufcmd(22,142,12,10,east);
252   2         call ufcmd(99,142,12,10,east);
253   2         call ufcmd(109,105,16,12,east);
254   2         call ufcmd(173,79,16,12,east);

                /* change color */
255   2         call umcmd(yellow,set);
                /* draw emergency fluid valves */
256   2         call ulcmd(24,150,31,143);
257   2         call ulcmd(101,150,108,143);

                /* draw reactor pump gear */
258   2         call ufcmd(115,107,4,8,east);
259   2         call ufcmd(113,109,8,4,east);
260   2         call ufcmd(116,106,2,10,east);
261   2         call ufcmd(112,110,10,2,east);

                /* draw steam generator pump gear */
262   2         call ufcmd(179,81,4,8,east);
263   2         call ufcmd(177,83,8,4,east);
264   2         call ufcmd(180,80,2,10,east);
265   2         call ufcmd(176,84,10,2,east);

                /* change color */
266   2         call umcmd(magenta,set);

                /* draw stationary elements inside reactor */
267   2         call ufcmd(53,124,5,14,east);
268   2         call ufcmd(63,124,5,14,east);
269   2         call ufcmd(73,124,5,14,east);

                /* change color */
270   2         call umcmd(cyan,set);

                /* draw moveable elements inside reactor */
271   2         call ufcmd(59,131,3,14,east);
272   2         call ufcmd(69,131,3,14,east);
273   2         call ulcmd(60,145,60,198);
274   2         call ulcmd(70,145,70,198);
```

```
                /********* FILL WITH FLUID *********/

                /* change color */
275   2         call umcmd(blue,set);

                /* fill reactor emergency fluid container */
276   2         call ufcmd(10,171,36,30,east);
277   2         call ufcmd(27,155,16,16,northeast);
278   2         call ufcmd(28,155,16,16,northeast);
279   2         call ufcmd(24,152,8,7,east);
280   2         call ufcmd(24,127,9,15,east);
281   2         call ufcmd(33,127,18,8,east);

                /* fill reactor */
282   2         call ufcmd(80,107,29,8,east);
283   2         call ufcmd(51,100,29,75,east);
284   2         call ufcmd(80,157,74,8,east);
285   2         call ufcmd(154,157,2,6,east);
286   2         call ufcmd(140,142,15,7,northeast);
287   2         call ufcmd(139,142,15,6,northeast);
288   2         call ufcmd(152,130,6,12,northeast);
289   2         call ufcmd(151,130,7,12,northeast);
290   2         call ufcmd(140,118,12,7,northeast);
291   2         call ufcmd(139,118,12,6,northeast);
292   2         call ufcmd(148,109,7,9,northeast);
293   2         call ufcmd(148,110,6,8,northeast);
294   2         call ufcmd(125,107,29,8,east);
295   2         call ufcmd(154,109,2,6,east);

                /* fill condenser pipe */
296   2         call ufcmd(200,143,88,8,east);
297   2         call ufcmd(198,143,2,6,east);
298   2         call ufcmd(214,128,6,15,northeast);
299   2         call ufcmd(213,128,7,15,northeast);
300   2         call ufcmd(205,119,9,7,northeast);
301   2         call ufcmd(205,120,8,6,northeast);
302   2         call ufcmd(200,117,88,8,east);
303   2         call ufcmd(198,119,2,6,east);

                /* fill steam generator emergency fluid container */
304   2         call ufcmd(87,171,36,30,east);
305   2         call ufcmd(89,169,32,2,east);
306   2         call ufcmd(91,167,28,2,east);
307   2         call ufcmd(93,166,24,1,east);
308   2         call ufcmd(101,152,8,4,east);
309   2         call ufcmd(101,127,9,15,east);
310   2         call ufcmd(110,127,20,8,east);

                /* fill steam generator liquid */
311   2         call ufcmd(192,100,32,12,east);
312   2         call ufcmd(204,81,8,19,east);
313   2         call ufcmd(189,81,15,9,east);
314   2         call ufcmd(142,81,31,8,east);
315   2         call ufcmd(142,89,9,11,east);
316   2         call ufcmd(130,100,32,76,east);

                /********* FILL WITH STEAM *********/

317   2      aa: y1 = 177;
318   2          k = 0;

319   2          do while y1 < 191;
320   3            call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
321   3            call ulcmd(130,y1,161,y1);
322   3            y1 = y1 + 2;
323   3            k = k + 1;
324   3          end;
```

```
325   2       bb: y1 = 191;

326   2           do while y1 < 197;
327   3             call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
328   3             call ulcmd(142,y1,149,y1);
329   3             y1 = y1 + 2;
330   3             k = k + 1;
331   3           end;

332   2       cc: y1 = 197;
333   2           x2 = 149;

334   2           do while y1 < 205;
335   3             call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
336   3             call ulcmd(142,y1,x2,y1);
337   3             y1 = y1 + 2;
338   3             x2 = x2 - 2;
339   3             k = k + 1;
340   3           end;

341   2       dd: x1 = 143;
342   2           y2 = 204;

343   2           do while x1 < 151;
344   3             call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
345   3             call ulcmd(x1,204,x1,y2);
346   3             x1 = x1 + 2;
347   3             y2 = y2 - 2;
348   3             k = k + 1;
349   3           end;

350   2       ee: x1 = 151;

351   2           do while x1 < 189;
352   3             call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
353   3             call ulcmd(x1,204,x1,197);
354   3             x1 = x1 + 2;
355   3             k = k + 1;
356   3           end;

357   2       ff: x1 = 189;
358   2           y2 = 197;

359   2           do while x1 < 197;
360   3             call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
361   3             call ulcmd(x1,204,x1,y2);
362   3             x1 = x1 + 2;
363   3             y2 = y2 + 2;
364   3             k = k + 1;
365   3           end;

366   2       gg: y1 = 203;
367   2           x2 = 196;

368   2           do while y1 > 195;
369   3             call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
370   3             call ulcmd(196,y1,x2,y1);
371   3             y1 = y1 - 2;
372   3             x2 = x2 - 2;
373   3             k = k + 1;
374   3           end;

375   2       hh: y1 = 195;

376   2           do while y1 > 191;
377   3             call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
```

```
378   3             call ulcmd(196,y1,189,y1);
379   3             y1 = y1 - 2;
380   3             k = k + 1;
381   3           end;
382   2       ii: y1 = 165;

383   2           do while y1 > 157;
384   3             call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
385   3             call ulcmd(204,y1,211,y1);
386   3             y1 = y1 - 2;
387   3             k = k + 1;
388   3           end;

389   2       jj: y1 = 157;

390   2           do while y1 > 151;
391   3             call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
392   3             call ulcmd(192,y1,223,y1);
393   3             y1 = y1 - 2;
394   3             k = k + 1;
395   3           end;

396   2       kk: call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
397   2           call ulcmd(192,151,201,151);
398   2           k = k + 1;

399   2       ll: y1 = 149;
400   2           x2 = 197;

401   2           do while y1 > 145;
402   3             call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
403   3             call ulcmd(192,y1,x2,y1);
404   3             y1 = y1 - 2;
405   3             x2 = x2 - 1;
406   3             k = k + 1;
407   3           end;

408   2       mm: y1 = 145;
409   2           x2 = 196;

410   2           do while y1 > 141;
411   3             call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
412   3             call ulcmd(192,y1,x2,y1);
413   3             y1 = y1 - 2;
414   3             x2 = x2 + 1;
415   3             k = k + 1;
416   3           end;

417   2       nn: y1 = 141;
418   2           x2 = 198;
419   2           x1 = 214;

420   2           do while y1 > 133;
421   3             call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
422   3             call ulcmd(192,y1,x2,y1);
423   3             call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
424   3             call ulcmd(x1,y1,223,y1);
425   3             y1 = y1 - 2;
426   3             x2 = x2 + 2;
427   3             x1 = x1 + 2;
428   3             k = k + 1;
429   3           end;

430   2       oo: y1 = 133;
431   2           x2 = 205;
432   2           x1 = 221;
```

```
433   2           do while y1 > 125;
434   3             call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
435   3             call ulcmd(192,y1,x2,y1);
436   3             call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
437   3             call ulcmd(x1,y1,223,y1);
438   3             y1 = y1 - 2;
439   3             x2 = x2 - 2;
440   3             x1 = x1 - 2;
441   3             k = k + 1;
442   3           end;

443   2       pp: y1 = 125;
444   2           x2 = 197;

445   2           do while y1 > 121;
446   3             call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
447   3             call ulcmd(192,y1,x2,y1);
448   3             y1 = y1 - 2;
449   3             x2 = x2 - 1;
450   3             k = k + 1;
451   3           end;

452   2       qq: y1 = 121;
453   2           x2 = 196;

454   2           do while y1 > 115;
455   3             call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
456   3             call ulcmd(192,y1,x2,y1);
457   3             y1 = y1 - 2;
458   3             x2 = x2 + 1;
459   3             k = k + 1;
460   3           end;

461   2       rr: y1 = 115;

462   2           do while y1 > 111;
463   3             call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
464   3             call ulcmd(192,y1,223,y1);
465   3             y1 = y1 - 2;
466   3             k = k + 1;
467   3           end;

                  /********* OUTPUT LABEL *********/

468   2           call strcmd(56,50,normal_char,east,two_x,red,
                              length(line1),@line1);
469   2           call strcmd(128,30,normal_char,east,two_x,blue,
                              length(line2),@line2);
470   2           call strcmd(0,10,normal_char,east,two_x,green,
                              length(line3),@line3);
                  /********* CAUSE MOTION *********/

                  /* initialize variable */
471   2           i = 0;

                  /* change write mode */
472   2           call umcmd(blue,complement);

473   2           do cycle = 0 to 2;

                    /* change line drawing pattern */
474   3             call agcmd(5555h,0ffffh,0ffffh,0ffffh);

475   3             i = ( (cycle and 3) / 2);
```

```
                    /* cause motion in reactor fluid */
476   3       aa1:  x1 = 108 - i;

477   3             do while x1 > 79;
478   4               call ulcmd(x1,107,x1,114);
479   4               x1 = x1 - 2;
480   4             end;

481   3       bb1:  y1 = 100 + i;

482   3             do while y1 < 175;
483   4               call ulcmd(51,y1,79,y1);
484   4               y1 = y1 + 2;
485   4             end;

486   3       cc1:  x1 = 80 + i;

487   3             do while x1 < 154;
488   4               call ulcmd(x1,157,x1,164);
489   4               x1 = x1 + 2;
490   4             end;

491   3       dd1:  x1 = 154 + i;

492   3             call ulcmd(x1,157,x1,162);

493   3       ee1:  x1 = 153 - (2 * i);
494   3             y1 = 155 - i;
495   3             x2 = 153 - (2 * i);
496   3             y2 = 155 - i;

497   3             do while y2 > 148;
498   4               call ulcmd(x1,y1,x2,y2);
499   4               x1 = x1 - 2;
500   4               x2 = x2 - 1;
501   4               y2 = y2 - 1;
502   4             end;

503   3       ff1:  x1 = 139;
504   3             y1 = 153 - i;
505   3             x2 = 134;
506   3             y2 = 148 - i;
507   3             j = i;

508   3             do while y2 > 129;
509   4               call ulcmd(x1,y1,x2,y2);
510   4               x2 = x2 + 1;
511   4               y2 = y2 - 1;
512   4               if (j and 1) = 0 then
513   4                 y1 = y1 - 2;
514   4               else
                        x1 = x1 + 2;
515   4               j = j + 1;
516   4             end;

517   3       gg1:  x1 = 145;
518   3             y1 = 134 - i;
519   3             x2 = 150;
520   3             y2 = 129 - i;
521   3             j = i;

522   3             do while y2 > 124;
523   4               call ulcmd(x1,y1,x2,y2);
524   4               x2 = x2 - 1;
525   4               y2 = y2 - 1;
526   4               if (j and 1) = 0 then
```

```
527   4                 y1 = y1 - 2;
528   4               else
                        x1 = x1 - 2;
529   4               j = j + 1;
530   4             end;

531   3     hh1:    x1 = 139;
532   3             y1 = 129 - i;
533   3             x2 = 134;
534   3             y2 = 124 - i;
535   3             j = i;

536   3             do while y2 > 106;
537   4               call ulcmd(x1,y1,x2,y2);
538   4               x2 = x2 + 1;
539   4               y2 = y2 - 1;
540   4               if (j and 1) = 0 then
541   4                 y1 = y1 - 2;
542   4               else
                        x1 = x1 + 2;
543   4               j = j + 1;
544   4             end;

545   3     ii1:    x1 = 148 - i;
546   3             y2 = 107 + i;

547   3             do while x1 > 140;
548   4               call ulcmd(x1,107,x1,y2);
549   4               x1 = x1 - 2;
550   4               y2 = y2 + 2;
551   4             end;
552   3     jj1:    x1 = 140 - i;

553   3             do while x1 > 124;
554   4               call ulcmd(x1,107,x1,114);
555   4               x1 = x1 - 2;
556   4             end;

                    /* cause motion in condenser pipe fluid */
557   3     aa2:    x1 = 287 - i;

558   3             do while x1 > 199;
559   4               call ulcmd(x1,143,x1,150);
560   4               x1 = x1 - 2;
561   4             end;

562   3     bb2:    x1 = 199 - i;

563   3             call ulcmd(x1,143,x1,148);

564   3     cc2:    x1 = 200 + (2 * i);
565   3             y1 = 141 - i;
566   3             x2 = 200 + (2 * i);
567   3             y2 = 141 - i;

568   3             do while y2 > 134;
569   4               call ulcmd(x1,y1,x2,y2);
570   4               x1 = x1 + 2;
571   4               x2 = x2 + 1;
572   4               y2 = y2 - 1;
573   4             end;

574   3     dd2:    x1 = 214;
575   3             y1 = 139 - i;
576   3             x2 = 219;
577   3             y2 = 134 - i;
```

```
578   3             j = i;

579   3             do while y2 > 116;
580   4               call ulcmd(x1,y1,x2,y2);
581   4               x2 = x2 - 1;
582   4               y2 = y2 - 1;
583   4               if (j and 1) = 0 then
584   4                 y1 = y1 - 2;
585   4               else
                        x1 = x1 - 2;
586   4               j = j + 1;
587   4             end;

588   3       ee2:  x1 = 205 + i;
589   3             y2 = 117 + i;

590   3             do while x1 < 213;
591   4               call ulcmd(x1,117,x1,y2);
592   4               x1 = x1 + 2;
593   4               y2 = y2 + 2;
594   4             end;
595   3       ff2:  x1 = 213 + i;

596   3             do while x1 < 288;
597   4               call ulcmd(x1,117,x1,124);
598   4               x1 = x1 + 2;
599   4             end;

                    /* cause motion in steam generator fluid */
600   3       aa3:  y1 = 111 - i;

601   3             do while y1 > 99;
602   4               call ulcmd(223,y1,192,y1);
603   4               y1 = y1 - 2;
604   4             end;

605   3       bb3:  y1 = 99 - i;

606   3             do while y1 > 89;
607   4               call ulcmd(211,y1,204,y1);
608   4               y1 = y1 - 2;
609   4             end;

610   3       cc3:  y1 = 89 - i;
611   3             x2 = 204 + i;

612   3             do while y1 > 81;
613   4               call ulcmd(211,y1,x2,y1);
614   4               y1 = y1 - 2;
615   4               x2 = x2 + 2;
616   4             end;

617   3       dd3:  x1 = 210 - i;
618   3             y2 = 81 + i;

619   3             do while x1 > 202;
620   4               call ulcmd(x1,81,x1,y2);
621   4               x1 = x1 - 2;
622   4               y2 = y2 + 2;
623   4             end;

624   3       ee3:  x1 = 202 - i;

625   3             do while x1 > 188;
626   4               call ulcmd(x1,81,x1,88);
```

```
627   4               x1 = x1 - 2;
628   4             end;

629   3       ff3:  x1 = 172 - i;

630   3             do while x1 > 150;
631   4               call ulcmd(x1,81,x1,88);
632   4               x1 = x1 - 2;
633   4             end;

634   3       gg3:  x1 = 150 - i;
635   3             y2 = 88 - i;
636   3             do while x1 > 142;
637   4               call ulcmd(x1,81,x1,y2);
638   4               x1 = x1 - 2;
639   4               y2 = y2 - 2;
640   4             end;

641   3       hh3:  y1 = 82 + i;
642   3             x2 = 142 + i;

643   3             do while y1 < 90;
644   4               call ulcmd(142,y1,x2,y1);
645   4               y1 = y1 + 2;
646   4               x2 = x2 + 2;
647   4             end;

648   3       ii3:  y1 = 90 + i;

649   3             do while y1 < 100;
650   4               call ulcmd(142,y1,149,y1);
651   4               y1 = y1 + 2;
652   4             end;

653   3       jj3:  y1 = 100 + i;

654   3             do while y1 < 106;
655   4               call ulcmd(130,y1,161,y1);
656   4               y1 = y1 + 2;
657   4             end;

658   3       kk3:  x1 = 154;
659   3             y1 = 106 + i;

660   3             do while y1 < 112;
661   4               call ulcmd(x1,y1,161,y1);
662   4               x1 = x1 + 2;
663   4               y1 = y1 + 2;
664   4             end;

665   3       ll3:  x1 = 158;
666   3             y1 = 112 + i;

667   3             do while y1 < 116;
668   4               call ulcmd(x1,y1,161,y1);
669   4               x1 = x1 - 2;
670   4               y1 = y1 + 2;
671   4             end;

672   3       mm3:  x1 = 154;
673   3             y1 = 116 + i;
674   3             x2 = 138;

675   3             do while y1 < 124;
676   4               call ulcmd(130,y1,x2,y1);
677   4               call ulcmd(x1,y1,161,y1);
```

```
678   4              x1 = x1 - 2;
679   4              y1 = y1 + 2;
680   4              x2 = x2 - 2;

681   4            end;

682   3     nn3:   x1 = 148;
683   3            y1 = 124 + i;
684   3            x2 = 132;

685   3            do while y1 < 136;
686   4              call ulcmd(130,y1,x2,y1);
687   4              call ulcmd(x1,y1,161,y1);
688   4              x1 = x1 + 2;
689   4              y1 = y1 + 2;
690   4              x2 = x2 + 2;
691   4            end;

692   3     oo3:   x1 = 158;
693   3            y1 = 136 + i;
694   3            x2 = 142;

695   3            do while y1 < 148;
696   4              call ulcmd(130,y1,x2,y1);
697   4              call ulcmd(x1,y1,161,y1);
698   4              x1 = x1 - 2;
699   4              y1 = y1 + 2;
700   4              x2 = x2 - 2;
701   4            end;

702   3     pp3:   x1 = 148;
703   3            y1 = 148 + i;
704   3            x2 = 132;

705   3            do while y1 < 156;
706   4              call ulcmd(130,y1,x2,y1);
707   4              call ulcmd(x1,y1,161,y1);
708   4              x1 = x1 + 2;
709   4              y1 = y1 + 2;
710   4              x2 = x2 + 2;
711   4            end;

712   3     qq3:   x1 = 156;
713   3            y1 = 156 + i;

714   3            do while y1 < 160;
715   4              call ulcmd(x1,y1,161,y1);
716   4              x1 = x1 + 2;
717   4              y1 = y1 + 2;
718   4            end;

719   3     rr3:   x1 = 158;
720   3            y1 = 160 + i;

721   3            do while y1 < 166;
722   4              call ulcmd(x1,y1,161,y1);
723   4              x1 = x1 - 2;
724   4              y1 = y1 + 2;
725   4            end;

726   3     ss3:   y1 = 166 + i;
727   3            do while y1 < 176;
728   4              call ulcmd(130,y1,161,y1);
729   4              y1 = y1 + 2;
730   4            end;
```

```
                    /* cause motion in steam */
731   3       as3:  y1 = 177;
732   3             k = 1;
733   3             l = l - 1;

734   3             do while y1 < 191;
735   4               call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
736   4               call ulcmd(130,y1,161,y1);
737   4               call agcmd( pattern((k - 1) and 3),0ffffh,0ffffh,0ffffh );
738   4               call ulcmd(130,y1,161,y1);
739   4               y1 = y1 + 2;
740   4               k = k + 1;
741   4             end;

742   3       bs3:  y1 = 191;

743   3             do while y1 < 197;
744   4               call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
745   4               call ulcmd(142,y1,149,y1);
746   4               call agcmd( pattern((k - 1) and 3),0ffffh,0ffffh,0ffffh );
747   4               call ulcmd(142,y1,149,y1);
748   4               y1 = y1 + 2;
749   4               k = k + 1;
750   4             end;

751   3       cs3:  y1 = 197;
752   3             x2 = 149;

753   3             do while y1 < 205;
754   4               call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
755   4               call ulcmd(142,y1,x2,y1);
756   4               call agcmd( pattern((k - 1) and 3),0ffffh,0ffffh,0ffffh );
757   4               call ulcmd(142,y1,x2,y1);
758   4               y1 = y1 + 2;
759   4               x2 = x2 - 2;
760   4               k = k + 1;
761   4             end;

762   3       ds3:  x1 = 143;
763   3             y2 = 204;

764   3             do while x1 < 151;
765   4               call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
766   4               call ulcmd(x1,204,x1,y2);
767   4               call agcmd( pattern((k - 1) and 3),0ffffh,0ffffh,0ffffh );
768   4               call ulcmd(x1,204,x1,y2);
769   4               x1 = x1 + 2;
770   4               y2 = y2 - 2;
771   4               k = k + 1;
772   4             end;
773   3       es3:  x1 = 151;

774   3             do while x1 < 189;
775   4               call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
776   4               call ulcmd(x1,204,x1,197);
777   4               call agcmd( pattern((k - 1) and 3),0ffffh,0ffffh,0ffffh );
778   4               call ulcmd(x1,204,x1,197);
779   4               x1 = x1 + 2;
780   4               k = k + 1;
781   4             end;

782   3       fs3:  x1 = 189;
783   3             y2 = 197;

784   3             do while x1 < 197;
785   4               call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
```

```
786   4              call ulcmd(x1,204,x1,y2);
787   4              call agcmd( pattern((k - 1) and 3),0ffffh,0ffffh,0ffffh );
788   4              call ulcmd(x1,204,x1,y2);
789   4              x1 = x1 + 2;
790   4              y2 = y2 + 2;
791   4              k = k + 1;
792   4            end;

793   3      gs3:  y1 = 203;
794   3            x2 = 196;

795   3            do while y1 > 195;
796   4              call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
797   4              call ulcmd(196,y1,x2,y1);
798   4              call agcmd( pattern((k - 1) and 3),0ffffh,0ffffh,0ffffh );
799   4              call ulcmd(196,y1,x2,y1);
800   4              y1 = y1 - 2;
801   4              x2 = x2 - 2;
802   4              k = k + 1;
803   4            end;

804   3      hs3:  y1 = 195;

805   3            do while y1 > 191;
806   4              call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
807   4              call ulcmd(196,y1,189,y1);
808   4              call agcmd( pattern((k - 1) and 3),0ffffh,0ffffh,0ffffh );
809   4              call ulcmd(196,y1,189,y1);
810   4              y1 = y1 - 2;
811   4              k = k + 1;
812   4            end;

813   3      is3:  y1 = 165;

814   3            do while y1 > 157;
815   4              call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
816   4              call ulcmd(204,y1,211,y1);
817   4              call agcmd( pattern((k - 1) and 3),0ffffh,0ffffh,0ffffh );
818   4              call ulcmd(204,y1,211,y1);
819   4              y1 = y1 - 2;
820   4              k = k + 1;
821   4            end;

822   3      js3:  y1 = 157;

823   3            do while y1 > 151;
824   4              call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
825   4              call ulcmd(192,y1,223,y1);
826   4              call agcmd( pattern((k - 1) and 3),0ffffh,0ffffh,0ffffh );
827   4              call ulcmd(192,y1,223,y1);
828   4              y1 = y1 - 2;
829   4              k = k + 1;
830   4            end;

831   3      ks3:  call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
832   3            call ulcmd(192,151,201,151);
833   3            call agcmd( pattern((k - 1) and 3),0ffffh,0ffffh,0ffffh );
834   3            call ulcmd(192,151,201,151);
835   3            k = k + 1;

836   3      ls3:  y1 = 149;
837   3            x2 = 197;

838   3            do while y1 > 145;
839   4              call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
840   4              call ulcmd(192,y1,x2,y1);
```

```
841   4                call agcmd( pattern((k - 1) and 3),0ffffh,0ffffh,0ffffh );
842   4                call ulcmd(192,y1,x2,y1);
843   4                y1 = y1 - 2;
844   4                x2 = x2 - 1;
845   4                k = k + 1;
846   4              end;

847   3      ms3:    y1 = 145;
848   3              x2 = 196;

849   3              do while y1 > 141;
850   4                call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
851   4                call ulcmd(192,y1,x2,y1);
852   4                call agcmd( pattern((k - 1) and 3),0ffffh,0ffffh,0ffffh );
853   4                call ulcmd(192,y1,x2,y1);
854   4                y1 = y1 - 2;
855   4                x2 = x2 + 1;
856   4                k = k + 1;
857   4              end;

858   3      ns3:    y1 = 141;
859   3              x2 = 198;
860   3              x1 = 214;

861   3              do while y1 > 133;
862   4                call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
863   4                call ulcmd(192,y1,x2,y1);
864   4                call agcmd( pattern((k - 1) and 3),0ffffh,0ffffh,0ffffh );
865   4                call ulcmd(192,y1,x2,y1);
866   4                call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
867   4                call ulcmd(x1,y1,223,y1);
868   4                call agcmd( pattern((k - 1) and 3),0ffffh,0ffffh,0ffffh );
869   4                call ulcmd(x1,y1,223,y1);
870   4                y1 = y1 - 2;
871   4                x2 = x2 + 2;
872   4                x1 = x1 + 2;
873   4                k = k + 1;
874   4              end;

875   3      os3:    y1 = 133;
876   3              x2 = 205;
877   3              x1 = 221;

878   3              do while y1 > 125;
879   4                call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
880   4                call ulcmd(192,y1,x2,y1);
881   4                call agcmd( pattern((k - 1) and 3),0ffffh,0ffffh,0ffffh );
882   4                call ulcmd(192,y1,x2,y1);
883   4                call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
884   4                call ulcmd(x1,y1,223,y1);
885   4                call agcmd( pattern((k - 1) and 3),0ffffh,0ffffh,0ffffh );
886   4                call ulcmd(x1,y1,223,y1);
887   4                y1 = y1 - 2;
888   4                x2 = x2 - 2;
889   4                x1 = x1 - 2;
890   4                k = k + 1;
891   4              end;

892   3      ps3:    y1 = 125;
893   3              x2 = 197;

894   3              do while y1 > 121;
895   4                call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
896   4                call ulcmd(192,y1,x2,y1);
897   4                call agcmd( pattern((k - 1) and 3),0ffffh,0ffffh,0ffffh );
898   4                call ulcmd(192,y1,x2,y1);
```

```
899   4                 y1 = y1 - 2;
900   4                 x2 = x2 - 1;
901   4                 k = k + 1;
902   4               end;

903   3       qs3:    y1 = 121;
904   3               x2 = 196;

905   3               do while y1 > 115;
906   4                 call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
907   4                 call ulcmd(192,y1,x2,y1);
908   4                 call agcmd( pattern((k - 1) and 3),0ffffh,0ffffh,0ffffh );
909   4                 call ulcmd(192,y1,x2,y1);
910   4                 y1 = y1 - 2;
911   4                 x2 = x2 + 1;
912   4                 k = k + 1;
913   4               end;

914   3       rs3:    y1 = 115;

915   3               do while y1 > 111;
916   4                 call agcmd( pattern(k and 3),0ffffh,0ffffh,0ffffh );
917   4                 call ulcmd(192,y1,223,y1);

918   4                 call agcmd( pattern((k - 1) and 3),0ffffh,0ffffh,0ffffh )
919   4                 call ulcmd(192,y1,223,y1);
920   4                 y1 = y1 - 2;
921   4                 k = k + 1;
922   4               end;
923   3             end;

924   2         end reactor;

                  /*****************************************************************

                                                      MAIN

                  *****************************************************************/

925   1         call init(4096,288,227,2,3,2,3,16,16,12h,82h,80h);
926   1         call reactor;
927   1         call dqexit(0);

928   1       end reactor$module;


MODULE INFORMATION:

     CODE AREA SIZE     = 2859H  10329D
     CONSTANT AREA SIZE = 0027H     39D
     VARIABLE AREA SIZE = 0000H      0D
     MAXIMUM STACK SIZE = 002EH     46D
     1268 LINES READ
     0 PROGRAM WARNINGS
     0 PROGRAM ERRORS

DICTIONARY SUMMARY:

     360KB MEMORY AVAILABLE
     11KB MEMORY USED   (3%)
     0KB DISK SPACE USED
```

June 1983

Computer Graphics Needs Standards As A Foundation For Future Growth

Miles Lewitt and Bruce Cohen
Intel Corporation

Order Number: 230722-001

Computer graphics needs standards as a foundation for future growth

Software and interface standards must pave the way for successive generations of VLSI to transform the face of computer displays

by Miles Lewitt and Bruce Cohen
Intel Corp., Hillsboro, Ore.

☐ Computer terminals are coming alive with graphics, displaying colorful images as varied as Disney animation or circuit schematics. But this transformation will be realized to its fullest extent only if computer graphics standards are generally adopted to enable different manufacturers from different countries to work in concert. New applications of computers are emerging even now, thanks to graphics, and more traditional applications are undergoing a metamorphosis as computer graphics comes into its own.

The advantages of mixing graphics and computers are obvious, from easier and friendlier user communication to the more immediate transmission of information that can be achieved only by pictures. Industry experts believe that by 1985 most computer-based systems will have some sort of graphics capability.

The technology and the recognition of computer graphics' advantages are over 20 years old. It is coming of age only now, however, because of the advent of the very large-scale integrated circuits that have made the requisite hardware affordable. VLSI presents the system designer with the opportunity to build graphics systems lower in cost, higher in performance and density, and with more reliability than ever before.

However, systems that are not designed to allow the incorporation of future technological advancements risk rapid obsolescence. Therefore, designers are challenged to create systems that can remain competitive by easily assimilating successive generations of VLSI

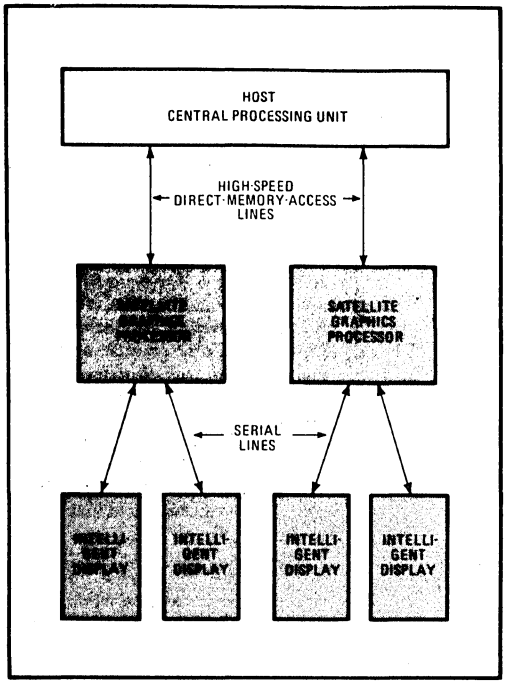
The widespread use of graphics will be hastened by timely acceptance of standards. Although the recent level of activity in graphics standards has heightened, the industry still needs wider participation, an understanding of the task's importance, and a spirit of cooperation.

In the last decade, many organizations and individuals have worked on standardizing computer graphics interfaces. Professional groups such as the Association for Computing Machinery have advocated and worked on the development of standards. Also, equally intent on realizing that goal, standards agencies such as the American National Standards Institute and the International Standards Organization have formed technical committees to develop graphic standards.[1,2]

On the move

Each of the groups working on graphics standards has been motivated differently, but all seek a standard interface for computer graphics. Portability is the prime need. For example, in application software, graphics users want their software to be portable from one host computer to another and from one operating system to another. Besides releasing users from a sole-source trap, portability also slows the obsolesence of software by allowing it to run on successive generations of computers. For the vendors of the hardware, it increases the available market for their products.

Another aspect of portability is device independence, or display portability. This allows a graphics application program to use any of a number of different display devices without "knowing" about each device's individual characteristics. Even similar technologies are not a prerequisite for the displays. For instance, a pen-plotter



1. Spread it around. A typical distributed graphics system today will probably have graphics functions in at least three levels of hierarchy. The application packages may be in the host, imaging, or viewing functions in satellite graphics processors and in device drivers and display controllers in intelligent display units.

and a raster-scanned cathode-ray-tube terminal could work with the software equally well. Device independence lets a graphics system be tailored for its application environment without a major effort aimed at rewriting existing software.

A third aspect of portability is image portability. Identical pictures reproduced on different devices should look as much alike as possible. Image portability makes a graphics system less hostile to someone with no computer expertise.

Standards for communicating and storing images are as important as those for generating them. If a standard exists for representing the contents of a picture, then the picture can be sent from one place to another by any means or be stored for a while and played back on a different device.

Two trends in the advancement of graphics systems architecture in the last few years have been distributed systems and the development of specialized subsystems. Graphic computations that a decade ago would have been performed in a host computer directly controlling a graphics display can now be done in a satellite processor. This processor can in turn be connected to an intelligent display device (Fig. 1).

Without a standard interface with graphics functions, each step in the development of distributed architectures has required the development of new software to allow the parts of the system to talk to each other. New generations of VLSI graphic-display controllers will result in yet another raft of software changes unless standards are adopted. With such standards in hand, systems can be upgraded by replacing an old generation of chips with a new generation having the same interface.

The great controversy

The graphics standard effort in the U. S. over the last 10 years has been intensive, but it is only now beginning to produce standards that the industry is likely to adopt. Great controversy has surrounded all attempts at standardization because standards must serve many application areas and system technologies.

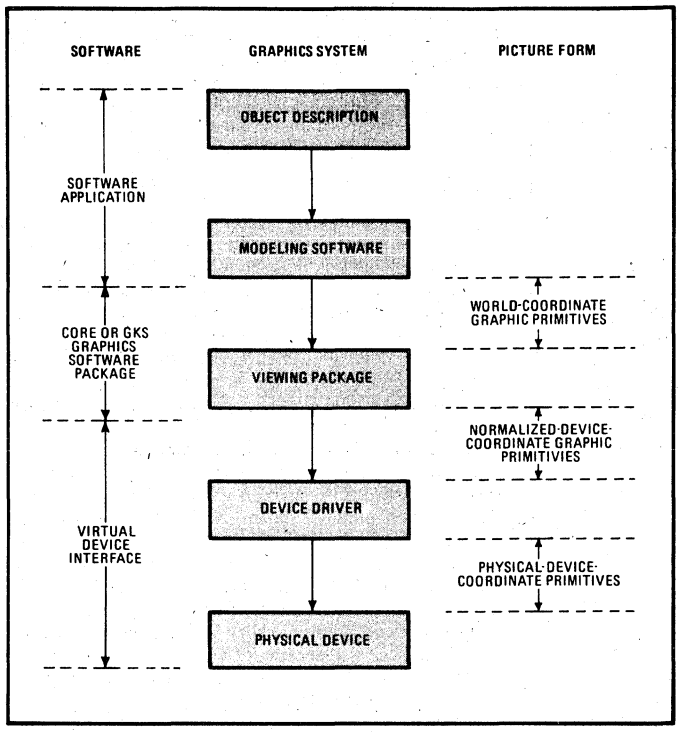
In the mid-1970s, the Special Interest Group on Graphics (Siggraph) of the ACM worked to develop a standard that would provide both two- and three-dimensional graphic functions that could be implemented with different levels of functionality, from simple to more sophisticated, in order to serve both large and small systems. In 1979 the Siggraph group called the Graphics Standards Planning Committee decided to hand over this proposed standard, called the GSPC Core, or simply Core, to ANSI to be developed into an American national standard. The ANSI X3H3 technical committee formed for this purpose included many of the Core developers. Attempts by vendors to market Core implementations designed to the 1979 Siggraph document have resulted in a number of interfaces—all, unfortunately, mutually incompatible to some degree.

Many graphics manufacturers and users have implemented the proposed Core standard anyway under the assumption that something approaching a standard is better than none at all. However, in what may be a breakthrough, the X3H3 committee on computer graphics voted overwhelmingly last October to seek formal approval of the European-developed Graphical Kernel System as a U. S. standard for device-independent graphics software. The executive committee of Siggraph is now investigating alternate methods of obtaining a recognized standard for the Core interface to provide standardization for the many users of existing Core packages.

GKS, a 2-d graphics interface, is similar in function to Core—which in addition encompasses 3-d graphics—but it is somewhat different in philosophy. Core was developed for a number of applications—computer-aided design and manufacturing being the main ones—in which the representation of an image is controlled by the user.

For example, employing a Core graphics software system, a user could set a line's color to be red and its style to be dashed. However, GKS's designers chose to emphasize distinguishability over control of representation. This means that GKS can ensure that two lines that must be distinguished will be represented differently but that the user may not be able to control how they differ. For instance, though the user can draw a red-dashed triangle and a blue solid-line circle with a Core system, he or she may try the same thing with a GKS system but the system may render these as a black wide solid-line triangle and black dashed thin-line circle.

Another difficulty for GKS is that many systems origi-



2. Pathway to pictures. From a description of an object in application software to a picture of that object on a display screen or on paper, a particular set of steps and processes are becoming the standard procedure in graphics systems.

nating in the U. S. are up and running with Core as a basis. However, one major difference is currently being ironed out, as work is under way in Europe to extend GKS to cover 3-d functions. Moreover, the ANSI X3H3 committee has expressed interest in helping or even leading this effort so that users who switch from Core to GKS will not lose functionality.

The Core task group of X3H3 is working on a much-extended standard called the Programmer's Hierarchical Interactive Graphics Standard (PHIGS). This standard is aimed at very sophisticated computer modeling applications employing graphics for realistic representations of objects. Though its development has just begun, the future of PHIGS may be brighter than that of Core because of its greater sweep.

With GKS as the likeliest candidate for a widespresd standard now, ANSI effort remains directed at making it an American standard and expanding it into a family of standards. Because it interfaces with only one level of the graphics hierarchy, the viewing-package level that turns the computer model into a picture (Fig. 2), the X3H3 committee is also working on the Virtual Device Interface (VDI) and Virtual Device Metafile (VDM) standards proposals, which address the interface at the device level. VDM, which ANSI will present to ISO for consideration as an international standard, is scheduled for public review in mid-1983. It deals with file-format standards for storing and transmitting a picture in terms of device-level primitives in the form of coordinates. VDI, the standard for converting the output of the viewing package into the physical-device primitives, will follow in six to nine months.

In addition, X3H3 is writing a standard for a Programmer's Minimal Interface to Graphics. PMIG is at the same level of the hierarchy as Core, but it is compatible at a minimal level with GKS. It is intended for use by programmers with little or no graphics expertise, in application areas such as business graphics or graphic arts. The draft American GKS and PMIG standards are scheduled to be available for public review by the middle of 1983.[2]

Then again, just to complicate the matter further, the intense interest in videotex, the interactive communication of text and graphics between central data bases and terminals, has resulted in a number of standards for the encoding of that information. The North American Presentation Level Protocol Syntax, the most ambitious yet, grew out of the Canadian Telidon system. Telidon, adopted by AT&T and enhanced as a standard for the transmission of text and graphic information, rose again as the North American standard.

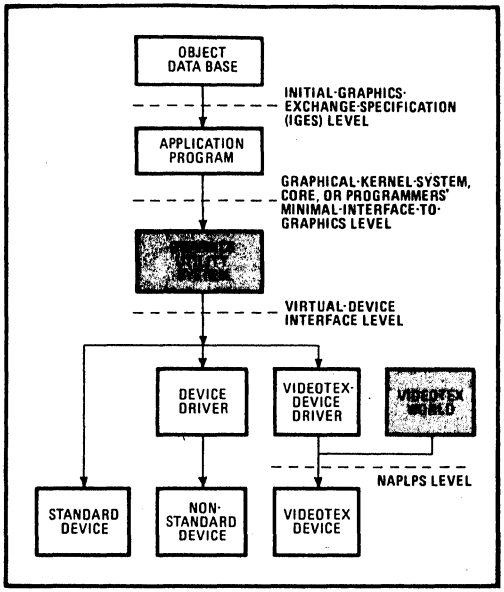
The videotex and teletext working group of the ANSI X3L2 technical committee on character codes, in cooperation with the Canadian Standards Association working group on videotex, has developed a technical specification with the AT&T standard as a baseline.[3] The document was accepted as a preliminary standard by CSA and was published for public review by ANSI in January 1983.

Some companies have already started investigating the possibility of expanding the functionality of the NAPLPS standard. Because one of the design goals for the standard was that the interface be extensible, it should be easy for users to add such functions. Those extensions that are the most widely applicable will no doubt present a good case to be added to the standard at a later date.

The primitive state

Graphics interface standards have been proposed at several levels of the hierarchy of a graphics system (Fig. 3). Application software deals with descriptions of the geometry and topology of the objects to be displayed. These objects are described in world coordinates which are the usual X, Y coordinates in 2-d space or the X, Y, Z coordinates in 3-d space.

In CAD applications, these descriptions would be of real objects, such as gears and bearings in 3-d space



3. Standards hierarchy. Current and proposed graphics standards specify interfaces at four levels of hierarchy in a typical graphics system. The IGES is a communications file structure for the exchange of data in computer-aided design systems. Other standards to which the hierarchy applies are GKS, CORE, PMIG, VDI, and NAPLPS.

using a standard set of high-level graphic primitives. In a business application, the description might be of points and lines on a chart, with the world coordinates as the units used for the chart.

The graphic primitives in world coordinates are transformed by the viewing package into a lower-level set of primitives expressed in what are called normalized or virtual-device coordinates, that is, coordinates for an idealized device. This transformation produces image data that is ready for a display device. For a particular device, a device driver will transform the virtual-device coordinates into commands for that device. If the world-coordinate space is 3-d, the viewing package might perform a perspective projection onto a viewing plane.

The proposed Core and GKS standards are examples of standard interfaces with the viewing package, which changes the description of a picture into a picture. Because of the standards' relatively high level in the system hierarchy and their inclusion of 3-d capability (in development for GKS), they are or will likely be extensively used in CAD and other sophisticated applications.

At the next level down, the virtual-device interface standard is intended as a single standard supporting as many types of graphics devices as possible with the same set of functions. The devices can range from pen-plotters to raster-scanned displays to microfilm recorders. The design philosophy of VDI is a balance between the simplicity desired for a low-level interface to allow it run efficiently in a variety of environments and the sophistication required to adequately support the expected range of higher-level graphics packages, that is, those that employ GKS or other higher-level standards. Its companion standard, the virtual-device metafile, or VDM, is being developed in parallel with VDI. The VDM is a standard means of storing or transmitting pictures.

Because VDI and VDM provide device-independent interfaces for display, storage, and transmission of graphic information, they do not exploit any one display technology to its full potential. Raster-scanned display, where dots are displayed line by line and appropriately turned on or off, is becoming the most widely used technology for graphic display because of the low cost of television-based displays and the use of VLSI—the cost per bit of VLSI memory continues to decrease and the performace and functionality per dollar of VLSI central processing units keeps going up. This trend is expected by most observers of the industry to continue in the next few years. It would therefore seem advantageous for the industry to adopt a standard interface especially suited to raster-scanning graphic information.

The NAPLPS is such a standard. It provides a common means of interchanging graphic information either by communication lines like telephones or satellites or by computer storage media such as floppy disks or magnetic-tape cartridges. It is designed for raster graphics devices and standardizes several methods of compressing the descriptions of graphic and pictorial information for efficiency. NAPLPS trades an easy method of modifying images plus a simple interface with interactive graphics systems based on other standards for efficient transmission and closeness of fit to raster devices.

Pulling together

A key element of Intel's strategy for graphics products is to adopt NAPLPS and VDI standards as the interface with functional graphics units. In conjunction with Digital Equipment Corp. and Tektronix Inc., Intel has announced its support for these graphics standards. After the original announcement, 12 additional companies have joined with Digital Equipment Corp., Intel, and Tektronix. Representing a wide range within different parts of the communications, computer, and graphics fields, they are Digital Research Inc., Graphics Software Systems Inc., Hazeltine Corp., ICL Ltd., ISSCO Graphics Inc., Mannesmann Tally Corp., Microsoft Inc., AEL Microtel Ltd., Norpak Ltd., Westinghouse Electric Corp., Xerox Corp., and Precision Visuals Inc.

The goals of the original three companies with regard to these graphics standards are:

- To ensure that the standards are technically sound.
- To assist in the rapid completion and adoption of the standards by contributing additional resources to support the ANSI committees.
- To encourage other vendors to adopt these graphics standards, for the success of the standards will depend on the breadth of product support they receive. □

References

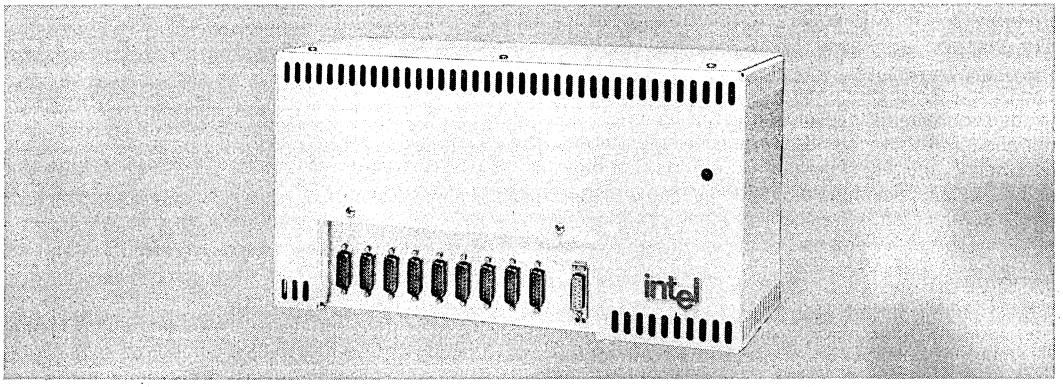
1. "Draft proposed American National Standard for the Virtual Device Metafile," ANSI X3H33 Virtual Device Interface Task Group, American National Standards Institute, Dec. 1982.
2. B. Shepherd, "Graphics Standards Status Report," Siggraph Computer Graphics, August 1979.
3. "Draft Standard, Videotex/Teletext Presentation Level Protocol Syntax (North American PLPS)," ANSI X3L2.1 Videotex Standing Task Group and CVCC/CSA Working Group on Videotex, June 1982.

Communications Controllers 10

iDCM 911-1 INTELLINK™ ETHERNET* CLUSTER MODULE

- **Eliminates need for transceivers and Ethernet coaxial cable for a local cluster of workstations**
- **Enables local cluster of nine workstations to connect to main Ethernet cable with only one transceiver**
- **Permits clustering of up to nine workstations in a smaller area**
- **Enables workstations to be up to 100M from main Ethernet cable**
- **Complies with the Ethernet Specification, Version 1.0, September 1980**

The Intellink™ Ethernet Cluster Module is a device used as a means of interconnecting up to nine Ethernet devices without the need for Ethernet coaxial cable and transceivers. The Intellink module forms a standalone Ethernet local area network with "interconnection" communication capability. The Intellink module (and attached devices) can optionally be connected to the Ethernet coaxial cable through a single transceiver



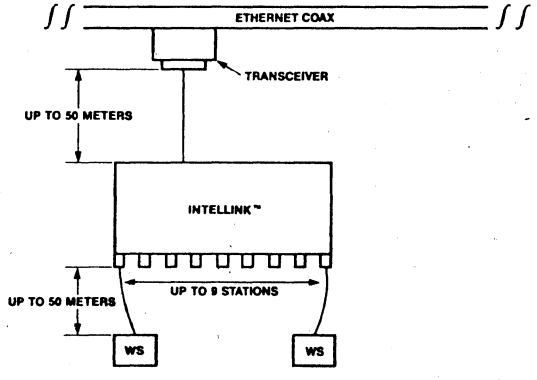


Figure 1. Intellink™ Configuration

* Ethernet is a trademark of Xerox Corporation.

ORDER NUMBER: 210508-001

FUNCTIONAL DESCRIPTION

Intellink module performs the same functions as a standard Ethernet transceiver. It buffers receive and transmit data, detects attempts by two or more stations to gain access to the line simultaneously, signals the presence of a collision to the transmitting stations, and transmits the jam signal prior to initiation of the random back-off algorithm. It complies with all of the interface parameters set forth in "The Ethernet Specification," 1.0 Version, September 1980.

Ethernet Work Station to Intellink™ Interface (WI) Connectors

There are nine WI interface connectors into which Ethernet-based systems can be connected. Each connector has the same signal pairs as does the equivalent connector on a standard Ethernet transceiver.

Intellink™ Module to Transceiver Interface (IT) Connector

The IT interface connector on the Intellink module is used to connect the local cluster to the "main" Ethernet cable through a standard transceiver, or can be left unconnected for standalone operation. The characteristics of this connector are identical to an Ethernet system to transceiver cable connector.

Topology

The Intellink module can function in standalone operation in which case it appears as a "zero length Ethernet segment" for up to nine Ethernet-based systems, or optionally can be connected to the "main" Ethernet coaxial cable through a single transceiver. When connected to the "main" Ethernet coaxial cable, it extends the Ethernet system interface to the transceiver from 50 meters to 100 meters. (Figure 1).

Physical Characteristics

Width ..... 14 in. (35.56 cm)
Height ..... 7.8 in. (19.81 cm)
Depth ..... 5.5 in. (13.97 cm)
Weight ..... 10 lb. (4.52 kg)

ELECTRICAL CHARACTERISTICS

Input Voltage Range:
(Voltages AC RMS)

Voltage (15%)
100V ±15%
120V ±15%
220V ±15%
240V ±15%

NOTE: The frequency range is 47 to 64 Hz, single phase.

ENVIRONMENTAL CHARACTERISTICS

Temperature: 10° to 40°C Operating
−40° to 70°C Non-Operating
Humidity: 10% to 85% Operating
5% to 95% Non-Operating

ORDERING INFORMATION

Part Number	Description
iDCM 911-1	Intellink, Ethernet cluster module, Version 1.0

iNA 960 NETWORK SOFTWARE

- **ISO Transport (8073) Class 4 services**
  - **—Guaranteed message integrity**
  - **—Data rate matching (flow control)**
  - **—Multiple connection capability**
  - **—Variable length messages**
  - **—Expedited delivery**
  - **—Negotiation of virtual circuit characteristics during opens**
- **Additional functionality**
  - **—Connectionless transport (Datagram)**
  - **—External Data Link**
- **IEEE 802.3 Data Link protocol (CSMA/CD) supported**
- **Comprehensive Network Management services**
  - **—Collection of network usage statistics**
  - **—Setting and inspecting of transport and data link parameters**
  - **—Fault isolation and detection**
  - **—Boot Server**
- **Compatible with multiple system environments**
  - **—Runs as an iRMX™ 86 job**
  - **—Supports host operating system independent designs based on 8086, 8088 or 80168 and 82586 components**
- **Runs on iSBC® 186/51 COMMputer™ Board**
- **Size configurable to suit specific application requirements**

iNA 960 is a general purpose local area network software package implementing the class 4 services of the ISO transport specification and network management functions in system designs based on the 8086, 8088 and 80186 microprocessors and the 82586 communications co-processor. iNA 960 also supports Intel's board level LAN products, the iSBC ® 550 KIT and the iSBC ® 186/51. Combined with the iSBC 186/51 COMMputer™ board, iNA 960 offers a high performance, cost effective network solution for MULTIBUS®/iRMX™ 86 users. See Figure 1 for iNA 960 functionality and operating environments.

iNA 960 is a ready-to-use software building block for OEM suppliers of networked systems for both technical and commercial applications. Examples for such applications include networked design stations, manufacturing process control, communicating word processors, and financial services workstations. Using the iNA 960 software the OEM can minimize development cost and time while achieving compatibility with a growing number of equipment suppliers adapting the IEEE and ISO standards.

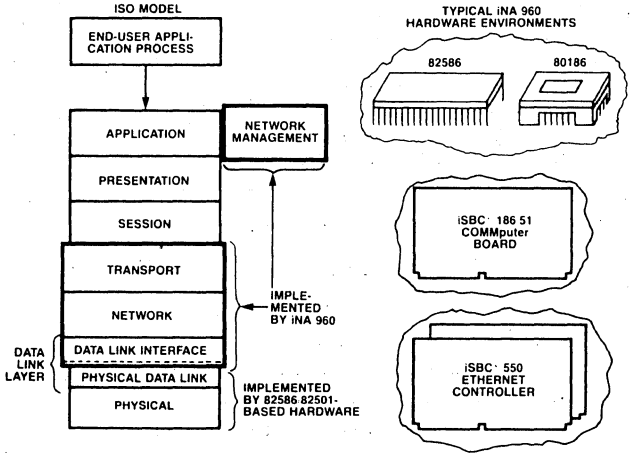


Figure 1.

MARCH 1984
ORDER NUMBER: 230777-002

FUNCTIONAL OVERVIEW

The iNA 960 design is a standard implementation of the Class 4 transport protocol defined by the ISO OSI model. The Transport Layer provides a reliable full-duplex message delivery service on top of the "best effort" IEEE 802.3 standard packet delivery service implemented by the 82586 (or equivalent) physical and data link functions.

Consisting of linkable modules, the software can be configured to implement a range of capabilities and interface protocols. In addition to reliable process-to-process message delivery, the capabilities include a datagram service, a boot server, a direct user access to the Data Link Layer, and a comprehensive network management facility.

iNA 960 can be configured to run under iRMX 86 along with the user software, or to run on top of a dedicated 8086, 8088 or 80186 processor coupled with an 82586 to provide a communications front end processor.

The software also includes a Network Management service. This facility enables the user to monitor and adjust the network's operation in order to optimize its performance.

The current release of iNA 960 includes a "null" Network Layer supporting the Data Link and Transport Layers without providing internetwork routing service. This capability will be implemented in later releases of iNA 960.

For a conceptual block diagram of iNA 960, refer to Figure 2.

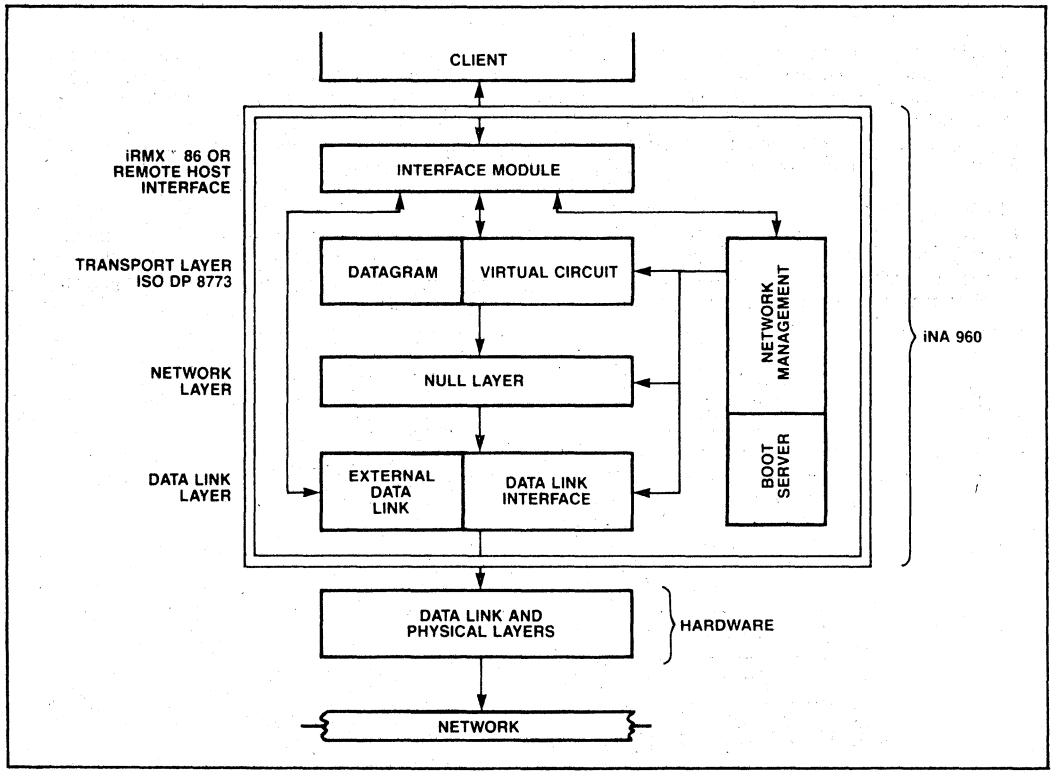


Figure 2. iNA 960 Conceptual Block Diagram

TRANSPORT LAYER

The Transport Layer provides message delivery services between client processes running on computers (network "hosts" or "nodes") anywhere in the network.

Client processes are identified by a combination of a network address defining the node and a transport service access point defining the interface point through which the client accesses the transport services. The combined parameters, called the transport address, are supplied by the user for both the local and the remote client processes to be connected.

The iNA 960 transport layer implements two kinds of message delivery services: virtual circuit and datagram. The virtual circuit provides a reliable point-to-point message delivery service ensuring maximum data integrity, and it is fully compatible with the ISO 8073 Class 4 protocol. The datagram service provides a best effort message delivery between client processes requiring less overhead and therefore allowing higher throughput than virtual circuits.

Both the datagram and the virtual circuit services are optional and can be included when configuring iNA 960.

Virtual Circuit Services

—Reliable Delivery: Data is delivered to the destination in the exact order it was sent by the source, with no errors, duplications or losses, regardless of the quality of service available from the underlying network service.

—Data Rate Matching (flow control): The Transport Layer attempts to maximize throughput while conserving communication subsystem resources by controlling the rate at which messages are sent. That rate is based on the availablity of receive buffers at the destination and its own resources.

—Multiple Connection Capability (Process Multiplexing): Several processes can be simultaneously using the Transport Layer with no risk that progress or lack of progress by one process will interfere with others.

—Variable Length Messages: The client software can submit arbitrarily short or long messages for transmittal without regard for the minimum or maximum network service data unit (NSDU) lengths supported by the underlying network services.

—Expedited Delivery (optional). With this service the client can transmit up to 16 bytes of urgent data bypassing the normal flow control. The expedited data is guaranteed to arrive before any normal data submitted afterward.

Connectionless Transport (Datagram) Service

The datagram service transfers data between client processes without establishing a virtual circuit. The service is a "best effort" capability and data may be lost or misordered. Data can be transferred at one time to a single destination or to several destinations (multicast).

NETWORK MANAGEMENT FACILITY (NMF)

The network management facility provides the users of the network with planning, operation, maintenance and initialization services described below.

—Planning: This service captures network usage statistics on the various layers to help plan network expansion. Statistics are maintained by the layers themselves and are made available to users via an interface with the NMF.

—Operation: This service allows the user to monitor network functions and to inspect and adjust network parameters. The goal is to provide the tools for performance optimization on the network.

—Maintenance: This service deals with detecting isolating and correcting network faults. It also provides the capability to determine the presence of hosts and the viability of their connection to the network.

—Initialization: NMF provides initialization and remote loading facilities.

Network management provides distributed management of the network; the user can request any of the services to be performed on a remote as well as a local node. The NMF interfaces to every other network layer both to utilize their services and to access their internal data bases.

In support of the above services, the NMF capabilities include layer management, echo testing, limited debugging facilities, and the ability to down line load and dump a remote system.

Layer management deals with manipulating the internal database of a layer. The elements of these data bases are termed objects. Some examples for objects are the number of collisions, retransmission time-out limit, the number of packets sent, and the list of nodes to boot. NMF can examine and modify objects in a layer's data base.

An echo facility is provided. Using this facility the host can determine if a node is present on the network or not, test the communication path to that node and determine whether the remote node is functional.

NMF enables the user to read or write memory in any host present on the network. This feature is provided as an aid to debugging.

NMF can down line load any system present on the network. A simple Data Link protocol is used to ensure reliability. This facility can be used to load databases, to boot systems without local mass storage or to boot a set of nodes remotely, thus ensuring that they have the same version of software, etc.

Dumping is an operation equivalent to memory read from the user's standpoint; however, dumping uses the Data Link facilities while memory read uses the transport facilities.

EXTERNAL DATA LINK (EDL)

The External Data Link option allows the user to access the functionalities of the Data Link Layer directly instead of having to go through the network and transport layers. This flexibility is useful when the user needs custom higher layer software, or does not need the Network Layer and Transport Layer services (e.g., when sending "best effort" messages, or running customer diagnostics).

Through the EDL the capabilities supporting the lower layers in iNA 960 are made directly available to the user. EDL enables the user to establish and delete data link connections, transmit packets to individual and multiple receivers, and configure the data link software to meet the requirements of the given network environment.

USER ENVIRONMENT

iNA 960 is designed to run on hardware based on the 8086, 8088 or 80186 microprocessors and the 82586 LAN Coprocessor. The software can be configured to run under iRMX 86 or on a dedicated 8086, 8088 or 80186 processor separately from the host. The following section describes these two operating environments.

iRMX Environment

In this configuration, both the user program and iNA 960 are running under iRMX 86. The communications software is implemented as an iRMX 86 job requiring the nucleus only for most operations. The only exception is the boot server option which also needs the Basic I/O System. iNA 960 will run in any iRMX environment including configurations based on the 80130. See Figure 3 for an illustration of iNA 960 running under iRMX 86.

Some of the typical hardware implementations include the iSBC 550 KIT combined with an 8086, 8088 or 80186 based host or the iSBC 186/51 COMMputer™ board integrating the host processor and the communications controller into a single, high performance MULTIBUS board. See Figure 4A and 4B for a conceptual block diagram of these configurations.

Operating System/Processor Independent Implementation

In those systems where iRMX 86 is not the primary operating system, where off-loading the host of the communications tasks is necessary for performance reasons, or where an existing communications front-end processor configuration is being upgraded, the user may wish to dedicate a processor for communications purposes. iNA 960 can be configured to support such implementations by providing network services on an 8086, 8088 or 80186 processor. Figure 5 depicts the conceptual block diagram of this configuration.

This approach provides the component and system designer with an ISO standard communications software building block that can be adapted to his system's needs with a minimum interfacing effort. For added flexibility, iNA 960 provides the user with the alternative of using the included interface module or writing his own module, if necessary.

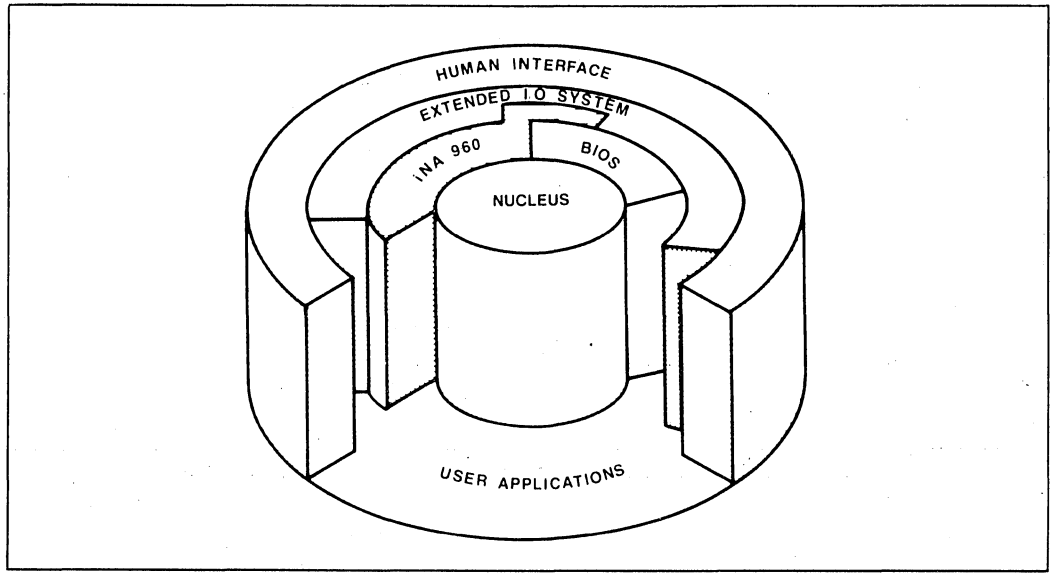


Figure 3. As an iRMX™ job, iNA 960 uses nucleus calls and, when the Boot Server is present, BIOS calls.

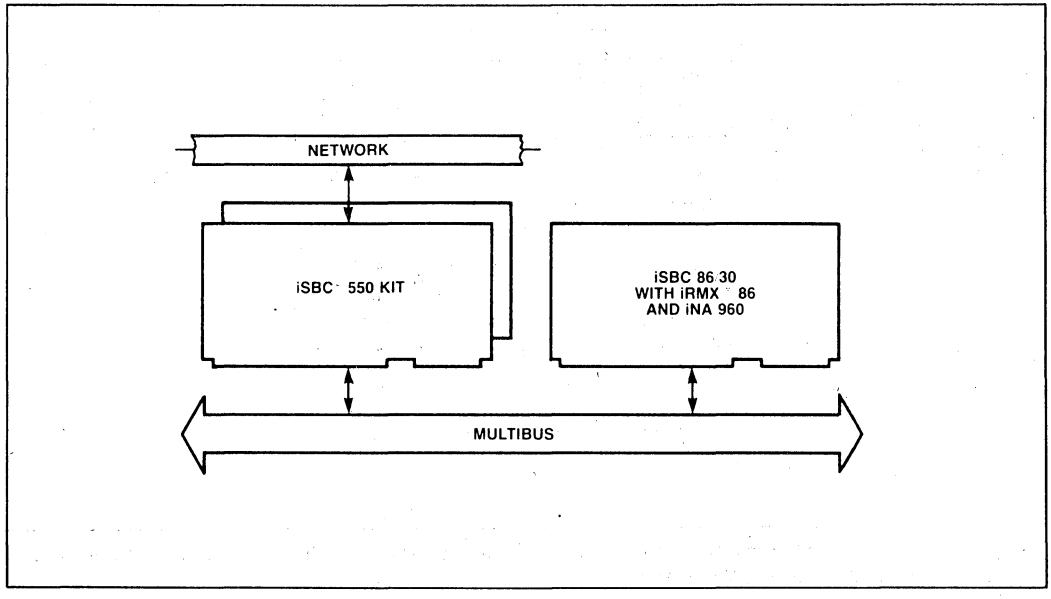


Figure 4A. Typical configuration using iSBC® 550 kit, iSBC® 86/30, iRMX 86™ and iNA 960.

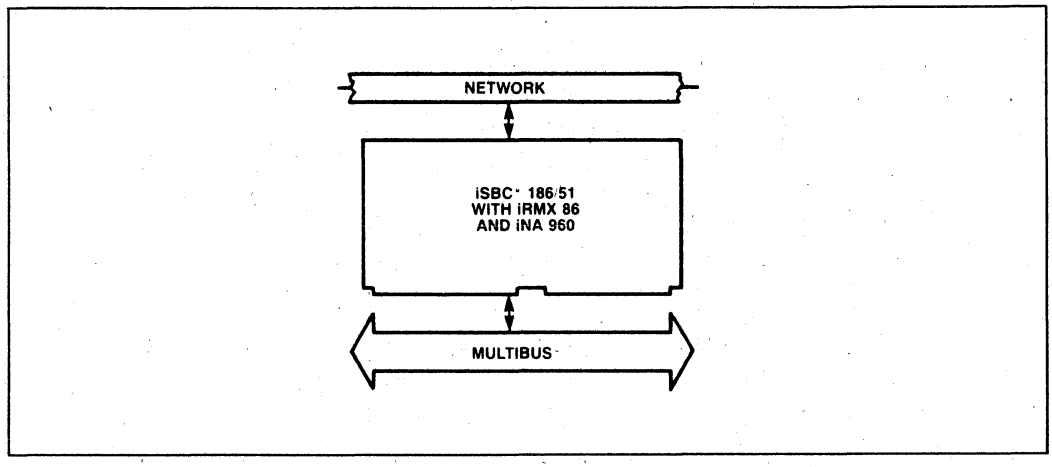


Figure 4B. Configuration using iSBC® 186/51, iRMX 86 and iNA 960.

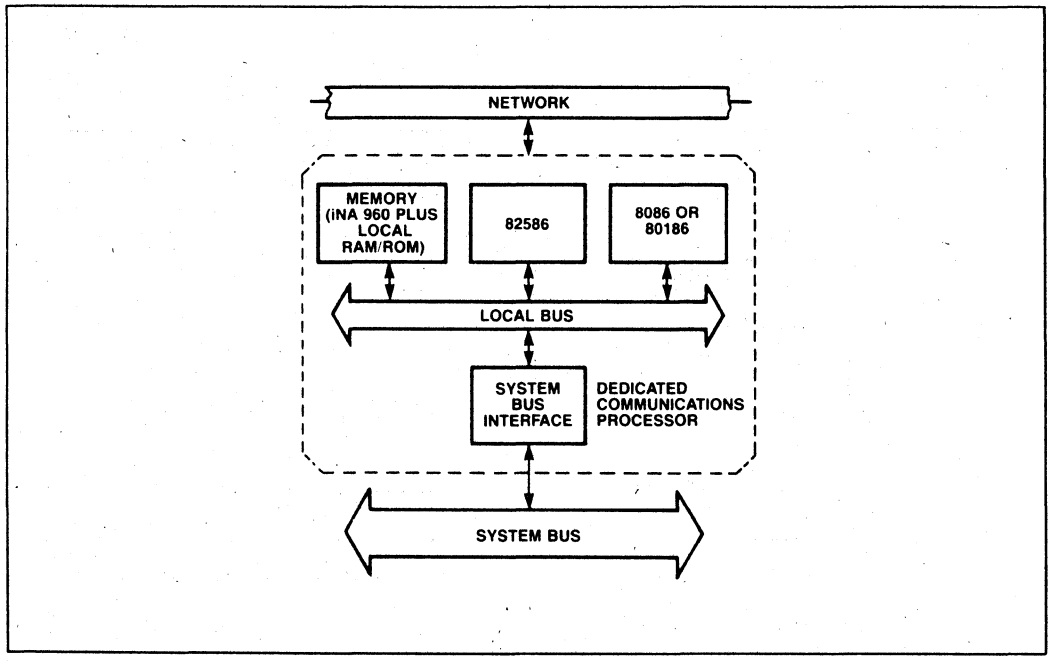


Figure 5. In the operating system/processor independent implementation iNA 960 is running on a dedicated 8086, 8088 or 80186 processor.

USER INTERFACE

iNA 960 is designed to run both under iRMX 86 and on a dedicated communications front end processor separately from the host. In both environments, the interface is based on exchanging memory segments called request blocks between iNA 960 and the client. The format and contents of the request blocks remain the same in both configurations; only the request block delivery mechanism changes. See Figure 6 for a simplified interface diagram.

Request blocks are memory segments containing the data to be passed from the user to iNA 960 (commands), or from iNA 960 to the user (responses). The iNA 960 request blocks consist of fixed format fields identical across all user commands and argument fields unique to the individual commands. Refer to Figure 7 for the standard request block format.

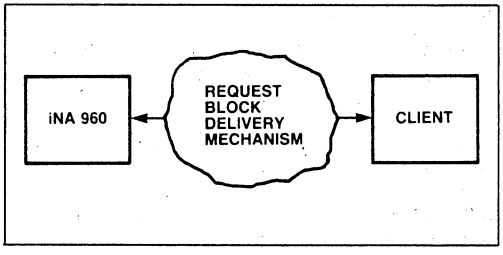


Figure 6.

Issuing an iNA 960 command consists of filling in the request block fields and transferring the block to iNA 960 for execution. After processing the command, iNA 960 returns the request block with one of the pre-defined response codes placed in the response code field of the request block. The response code indicates whether the command was executed successfully or whether an error occurred. By examining the response code, the user can take appropriate action for that command.

For iRMX users, iNA 960 also provides a procedural interface option to simplify writing the application software interface. In this case, the allocation and formatting of request blocks are replaced by a procedure call with parameters that specify the user's command options. The procedure execution will create a request block and fill in the appropriate fields from the user's parameter list.

For component users the request block delivery mechanism is the means by which the host processor and the communications processor running iNA 960 software exchange the request blocks. iNA 960 provides three such mechanisms: the MIP (Multibus Inter-process Protocol), the BCB (Base Control Block) and a user-defined mechanism. The MIP interface is included for use in systems already supporting this protocol; the BCB is a simple interface for single host environments, and the user-defined interface accommodates unique application requirements.

FIELDS	WORD/BYTE	
Reserved (2)	WORD	FIXED FORMAT FIELDS (same for all commands)
Length	BYTE	
User I.D.	WORD	
Response Port	BYTE	
Return Mailbox Token	WORD	
Segment Token	WORD	
Subsystem	BYTE	
Opcode	BYTE	
Response Code	WORD	
Arguments	BYTE	ARGUMENTS (changes by command)
•	•	
•	•	
•	•	

Figure 7. iNA 960 Request Block Format

Transport Layer User Interface

The following table summarizes the user commands and the corresponding transport layer responses.

Command	Function
1. OPEN	Allocates memory for the connection data base of a virtual circuit (or connection) to be established. The connection database contains data concerning the connection.
2. SEND CONNECT REQUEST	Requests connection to a fully specified remote transport address using specified ISO connection negotiation options.
3. AWAIT CONNECT REQUEST TRAN	Indicates that the transport client is willing to consider incoming connection requests based on pre-established acceptance criteria.
4. AWAIT CONNECT REQUEST USER	Indicates that the transport client is willing to consider incoming connection requests. If the request meets the address and negotiation option criteria, it is passed to the client for further consideration.
5. ACCEPT CONNECT REQUEST	Indicates that the connection requested by a remote transport service is accepted by the client.
6. SEND DATA or SEND EOM DATA	With this command, the client requests the transmission of the data in the buffers using the normal delivery service of the specified connection. The SEND EOM DATA command signals that the end of the data marks the end of the transport service data unit.
7. RECEIVE DATA	Posts normal receive data buffers for a specific connection or for a buffer pool used by a class of connections.
8. SEND EXPEDITED DATA	Transmits up to 16 bytes of data using the expedited delivery service. The expedited data is guaranteed to arrive at the destination before any normal data submitted afterward.
9. RECEIVE EXPEDITED DATA	Posts receive data buffers for expedited delivery for a specific connection or for a pool of buffers used by a class of connections.
10. CLOSE	Terminates an existing connection or rejects an incoming connection request. Any normal or expedited data queued up to be sent will not be sent.
11. AWAIT CLOSE	Requests notification of the client of the termination of a specified connection.
12. SEND DATAGRAM	Requests transmission of the data in the buffers using the transport datagram service.
13. RECEIVE DATAGRAM	Posts a receive buffer for a specific receiver or a class of receivers to receive data from a transport datagram.

Network Management Layer User Interface

Command	Function
1. READ OBJECT	Returns the value of the specified object to the client.
2. SET OBJECT	Sets the value of an object as specified by the client.
3. READ AND CLEAR OBJECT	Returns the value of the specified object to the client then clears the object.
4. ECHO	This function is used to determine the presence of a node, to test the communication path to the node and to ascertain the viability and functionality of the remote host addressed.
5. UP LINE DUMP	Requests a remote node to dump a specified memory area.
6. READ MEMORY	Reads memory of the specified network node.
7. SET MEMORY	Sets memory of the specified network node.
8. FORCE LOAD	Causes a node to attempt a remote load from another node.

External Data Link Interface

Command	Function
1. CONNECT	With this command the client establishes a data link connection.
2. DISCONNECT	Eliminates a previously established connection.
3. TRANSMIT	Transmits data contained in buffers specified by the client.
4. POST RECEIVE PACKET DESCRIPTOR	Allocates memory for maintaining records on receive data buffers. Also may be used to allocate memory for buffering receive data.
5. POST RECEIVE BUFFER	Allocates memory for buffering receive data.
6. ADD MULTICAST ADDRESS	Adds an address to the list of data link multicast addresses.
7. REMOVE MULTICAST ADDRESS	Removes an address from the list of data link multicast addresses.
8. SET DATA LINK I.D.	Sets up a unique data link I.D. for the station.

CONFIGURING iNA 960

In order to adapt iNA 960 to his specific application, the user must configure the software to define the desired functions, to select the appropriate interface, to set the layer parameters and to set up for the required hardware configuration.

There are a number of capability combinations the user may elect to implement in his application. At the transport layer level the options are: virtual circuit service with or without expedited delivery, or datagram service, or both. At the data link level, the user may include or exclude the External Data Link interface.

The Network Management Facility is also optional. When it is configured in, the user may also include the boot server module. These capabilities can be made available simply by linking in the corresponding software modules. The interface options are also implemented in a modular fashion; the user links in the desired module to set up for the iRMX 86 or the operating system independent configurations.

Layer parameters and confiuration options are first edited into layer configuration files, then assembled and linked into iNA 960. Layer parameters adjust the network's operation to match the usage pattern and the available resources. For example, within the Transport Layer, the flow control parameters, the retransmission timer parameters, the transport data base parameters, etc. can be set via this process.

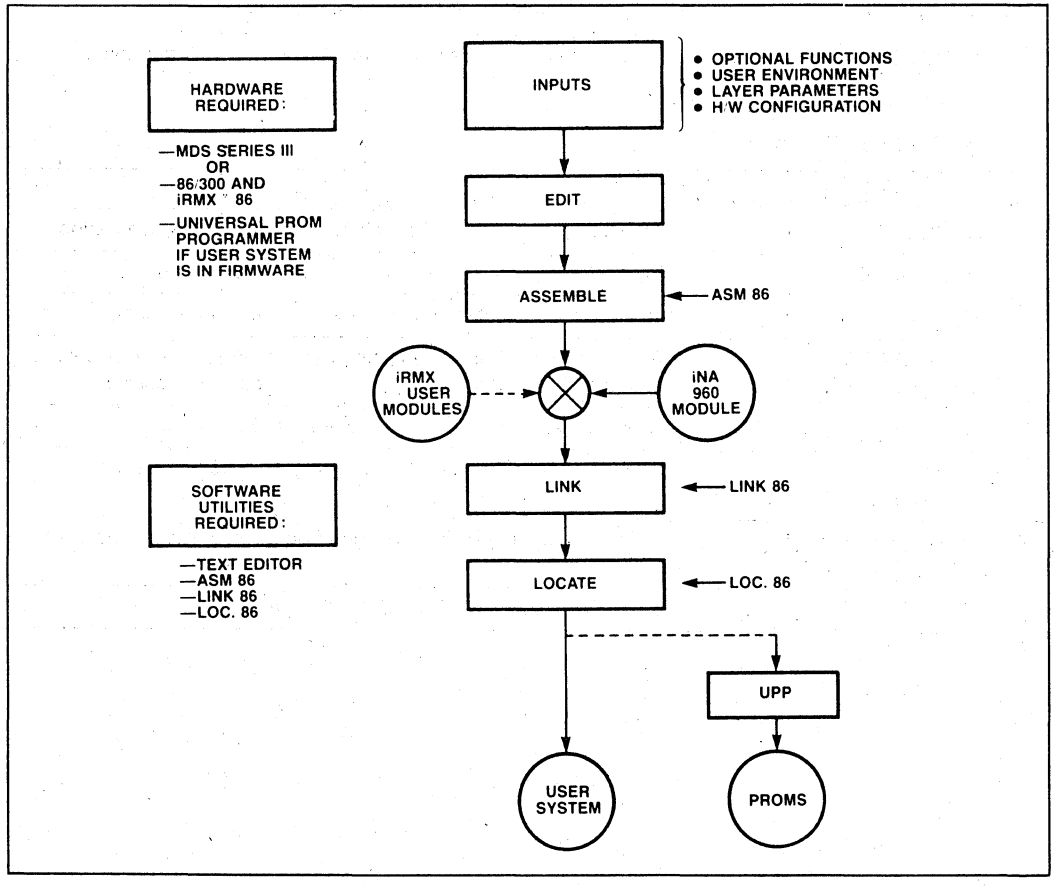


Figure 8. The Configuration Process for iNA 960

The user also sets up for the required hardware configuration, such as port addresses and interrupt levels, during this process. For the flow diagram of configuring iNA 960, refer to Figure 8.

SPECIFICATIONS

Hardware Supported:

—iSBC 186/51 Communicating Computer.
—iSBC 550 KIT Ethernet controller board(s) configured to run with iSBC 86/30 or iSBC 86/12B Multibus processor boards.
—Custom designs based on 8086, 8088 and 80186 microprocessors and the 82586 Local Communications Controller.

Typical Throughput at transport:

Environments:

186/51 and iRMX 86	50K to 200K bytes/sec
Dedicated 80186/ 82586 COMMengine	100K to 300K bytes/sec

Memory Requirements: (in bytes)

Base System	12K plus configurable Buffer Memory
Normal Virtual Circuit Option	18K plus configurable Buffer Memory
Expedited Delivery Option	2K
Datagram Option	3K plus Data Base Memory
Net Management Option	1K to 5K
External Data Link Option	5K
Boot Server Option	5K

Available Literature/ Reference Materials:

—iNA 960 Programmer's Reference Manual (11/83)
—iSBC 186/51 Data Sheet (Now)
—iSBC 186/51 Hardware Reference Manual (11/83)

Ordering Information

The following is a list of ordering options for the iNA 960 Network Software. All options include a full year of update service that provides a periodic NEWSLETTER, Software Problem Report Service, and copies of system updates that occur during this period. All of the object code options listed are available on either ISIS or RMX compatible double density diskettes.

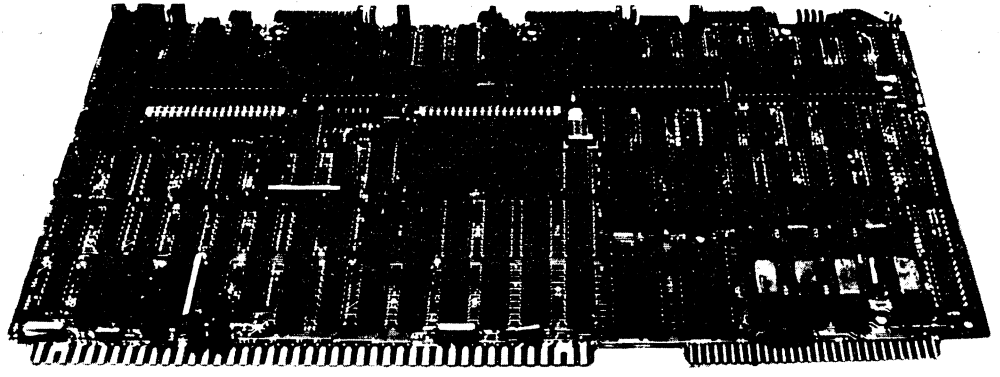
As with all Intel software, purchase of any of these options requires the execution of a standard Intel Master Software License. The specific rights granted to users depend on the specific option and the License signed.

Order Code	Description
iNA 960 YRO	OEM object code license requiring the payment of incorporation fees for each derivative work based on iNA 960; ISIS and RMX formatted diskettes
iNA 960 YST	Object code license to use the product at a second site or facility; ISIS and RMX formatted diskettes
iNA 960 YBY	Object code buy-out license requiring no further payment of incorporation fees; ISIS and RMS formatted diskettes
iNA 960 YSU	Object code single use license only; ISIS and RMS formatted diskettes
iNA 960 ESR	License for machine readable source code of iNA 960. RMX formulated diskettes.
iNA 960 LST	Source listing of iNA 960 provided on microfiche under a special source code license agreement
iNA 960 RF	Order code for the payment of incorporation fees

iSBC® 88/45
ADVANCED DATA COMMUNICATIONS PROCESSOR BOARD

- **Three HDLC/SDLC half/full-duplex communication channels — optional ASYNC/SYNC on two channels**
- **Supports RS232C (including modem support), CCITT V.24, or RS422A/449 interfaces**
- **On-board DMA supports 800K baud operation**
- **Self-clocking NRZI SDLC loop data link interface**
  - **— point-to-point**
  - **— multidrop**
- **Software programmable baud rate generation**
- **iAPX 88/10 (8088-2) Microprocessor operates at 8 MHz**
- **iSBC® 337 Numeric Data Processor option supported**
- **16K bytes static RAM (12K bytes dual-ported)**
- **Four 28-pin JEDEC sites for EPROM/RAM expansion; four additional 28-pin JEDEC sites added with iSBC® 341 board**
- **Two iSBX™ bus connectors**
- **MULTIBUS® interface supports Multimaster configuration**

The iSBC 88/45 Advanced Data Communications Processor (ADCP) Board adds 8 MHz, iAPX 88/10 (8088-2) 8-bit microprocessor-based communications flexibility to the Intel line of OEM microcomputer systems. The iSBC 88/45 ADCP board offers asynchronous, synchronous, SDLC, and HDLC serial interfaces for gateway networking or general purpose solutions. The iSBC 88/45 ADCP board provides the CPU, system clock, EPROM/RAM, serial I/O ports, priority interrupt logic, and programmable timers to facilitate higher-level application solutions.



OCTOBER 1984
ORDER NUMBER: 210372-002

FUNCTIONAL DESCRIPTION

Three Communication Channels

Three programmable HDLC/SDLC serial interfaces are provided on the iSBC 88/45 ADCP board. The SDLC interface is familiar to IBM system and terminal equipment users. The HDLC interface is known by users of CCITT's X.25 packet switching interface.

One channel utilizes an Intel 8273 controller to manage the serial data transfers. Accepting the 8-bit data bytes from the local bus, the 8273 controller translates the data into the HDLC/SDLC format. The channel operates in half/full-duplex mode.

In addition to the synchronous mode, the 8273 controller operates asynchronously with NRZI encoded data which is found in systems such as the IBM 3650 Retail Store System. An SDLC loop configuration using iSBX 352 and iSBC 88/45 products is shown in Figure 1.

The two additional channels utilize the Intel 8274 Multi-Protocol Serial Controller (MPSC). The MPSC provides two independent half/full-duplex serial channels which provide asynchronous, synchronous, HDLC or SDLC protocol operations. The sync and async protocol operations are commonly used to communicate with inexpensive terminals and systems.

The three serial channels of the iSBC 88/45 ADCP board offer communications capability to manage a gateway application. The gateway application, as shown in Figure 1, manages diverse protocol requirements for data movement between channels. Typical protocol management software layers implemented by the user include SNA terminal interfaces to IBM systems.

On-Board DMA

For high-speed communications, one MPSC channel has a DMA capacity to support an 800K baud rate. The second channel attached to the MPSC is capable of simultaneous 800K baud operation when configured with DMA capability, but is connected to an RS232C interface which is defined as 20K baud maximum. Figure 2 shows an RS422A/449 multidrop application which supports high-speed operation.

Interfaces Supported

The iSBC 88/45 ADCP board provides an excellent foundation to support these electrical and diverse software drivers protocol interfaces. The control lines, serial data lines, and signal ground lines are brought out to the three double-edge connectors. Figure 3 shows the cable to connector construction. Two connectors are pre-configured for RS422A/449. All three channels are configurable for RS232C/CCITT V.24 interfaces as shown in Table 1.

Table 1. iSBC® 88/45 Supported Configurations

Connection	Synchronous		Asynchronous	
	Modem	Direct	Modem*	Direct
point-to-point	X**	X	X	X
multidrop	X	X	X	X
loop	N.A.	N.A.	C (only)	C (only)

* Modem should not respond to break.
** Channels A, B, and C denoted by X.

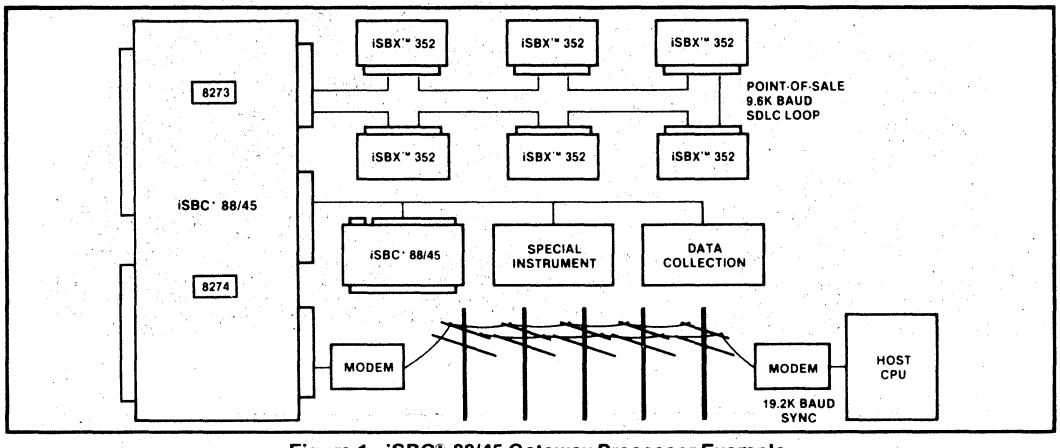
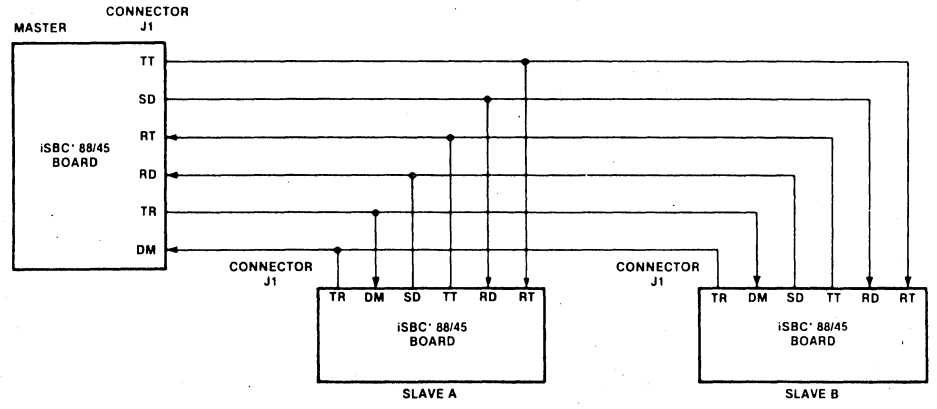


Figure 1. iSBC® 88/45 Gateway Processor Example



NOTE: The last slave device in the system must contain termination resistors on all signal lines received by the slave board. The master device contains bias resistors on all signal lines.

Figure 2. Synchronous Multidrop Network Configuration Example - RS422A

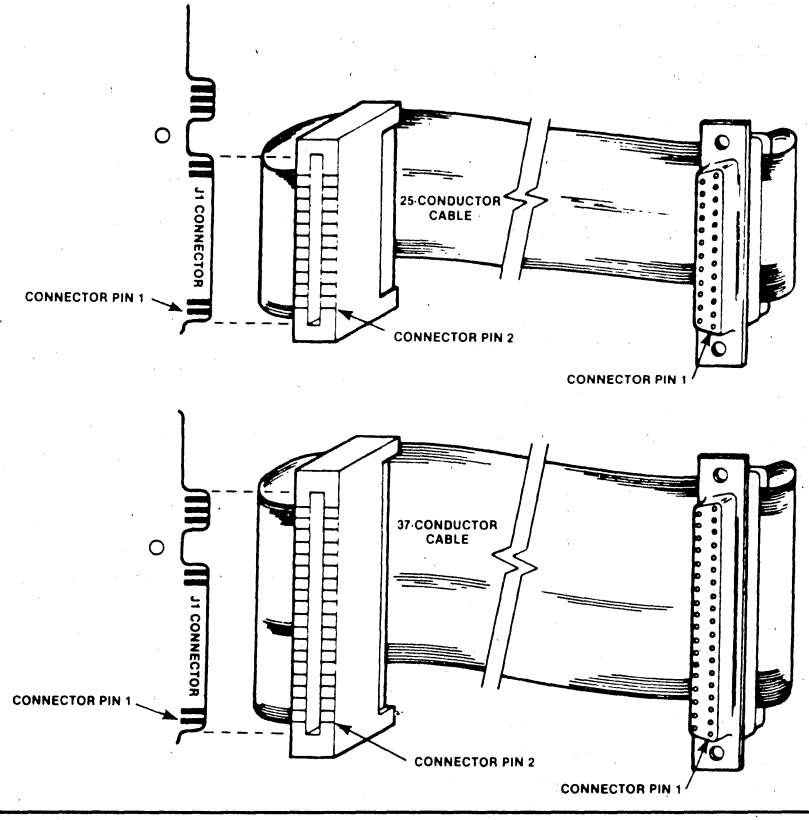


Figure 3. Cable Construction and Installation for RS232C and RS422A/449 Interface

Self Clocking Point-To-Point Interface

The iSBC 88/45 ADCP board is used in an asynchronous mode interface when configured as shown in Figure 4. The point-to-point RS232C example uses the self-clocking mode interface for NRZI encoding/decoding of data. The digital phase-lock loop allows operation of the interface in either half/duplex or full/duplex implementation with or without modems.

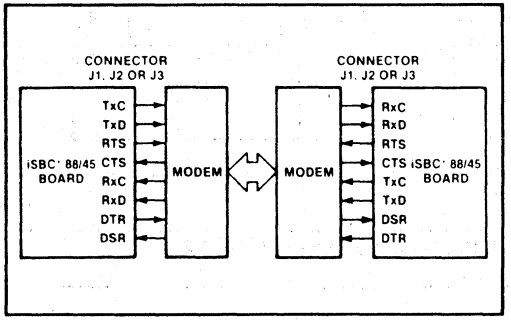


Figure 4. Self-Clocking or Asynchronous Point-to-Point Modem Interface Configuration Example - RS232C

Synchronous Point-To-Point Interface

Figure 5 shows a sychronous point-to-point mode of operation for the iSBC 88/45 ADCP board. This RS232C example uses a modem to generate the receive clock for coordination of the data transfer. The iSBC 88/45 ADCP board generates the transmit sychronizing clock for synchronous transmission.

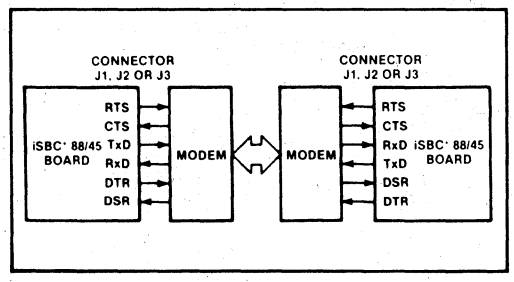


Figure 5. Synchronous Point-to-Point Modem Interface Configuration Example - RS232C

Central Processing Unit

The central processor for the iSBC 88/45 Advanced Data Communications Processor board is Intel's 8088 microprocessor operating at 8 MHz. The microprocessor interface to other functions is illustrated in Figure 6. The microprocessor architecture is designed to effectively execute the application and networking software written in higher-level languages.

This architectural support includes four 16-bit byte addressable data registers, two 16-bit memory base pointer registers and two 16-bit index registers. These registers are addressable through 24 different operand addressing modes for comprehensive memory addressing and for high-level language data structure manipulation.

The stack-oriented architecture readily supports Intel's iRMX executives and iMMX multiprocessing software. Both software packages are designed for modular application programming. Facilitating the fast inter-module communications, the 4-byte instruction queue supports program constructs needed for real-time systems.

Since programs are segmented between pure procedure and data, four segment registers (code, stack, data, extra) are available for addressing 1 megabyte of memory space. These registers contain the offset values used to address a 64K byte segment. The registers are controlled explicitly through program control or implicitly by high-level language functions and instructions.

The real-time system software can also utilize the programmable timers as shown in Table 2 and various interrupt control modes available on the ADCP board to have responsive and effective application solutions.

Table 2. Programmable Timer Functions

Function	Operation
Interrupt on Terminal Count	An interrupt is generated on terminal count being reached. This function is useful for generation of real-time clocks.
Rate Generator	Divide by N counter. Based on the input clock period, the output pulse remains low until the count is expired.
Square Wave Generator	Output remains high for one-half the count, goes low for the remainder of the count.
Software Triggered Strobe	Output remains high until count expires, then goes low for one clock period.

Numeric Data Processor Extension

The 8088 instruction set includes 8-bit and 16-bit signed and unsigned arithmetic operators for bi-

nary, BCD, and unpacked ASCII data. For enhanced numerics processing capability, the iSBC 337 MULTIMODULE Numeric Data Processor extends the 8088 architecture and data set[1].

The extended numerics capability includes over 60 numeric instructions offering arithmetic, trigonometric, transcendental, logarithmic, and exponential instructions. Many math-oriented applications utilize the 16-, 32-, and 64-bit integer, 32- and 64-bit floating point, 18-digit packed BCD, and 80-bit temporary data types.

16K Bytes Static Ram

The iSBC 88/45 ADCP board contains 16K bytes of high-speed static RAM, with 12K bytes dual-ported which is addressable from other MULTIBUS devices. When coupled with the high-speed DMA capability of the iSBC 88/45 ADCP board, the dual-ported memory provides effective data communication buffers. The dual-ported memory is useful for interprocessor message transfers.

[1] The iSBC 337 board requires the iSBC 88/45 ACDP board be jumpered to provide 4 MHz operation.

Interrupt Capability

The iSBC 88/45 ADCP board provides nine vectored interrupt levels. The highest level is the NMI (Non-Maskable Interrupt) line. The additional eight interrupt levels are vectored via the Intel 8259A Programmable Interrupt Controller (PIC). As shown in Table 3, four priority processing modes are available to match interrupt servicing requirements. These modes and priority assignments are dynamically configurable by the system software.

Table 3. Programmable Interrupt Modes

Mode	Operation
Nested	Interrupt request line priorities fixed; interrupt 0 is the highest and 7 is the lowest.
Auto-Rotating	The interrupt priority rotates; once an interrupt is serviced it becomes the lowest priority.
Specific Priority	System software assigns lowest level priority. The other levels are sequenced based on the level assigned.
Polled	System software examines priority interrupt via interrupt status register.

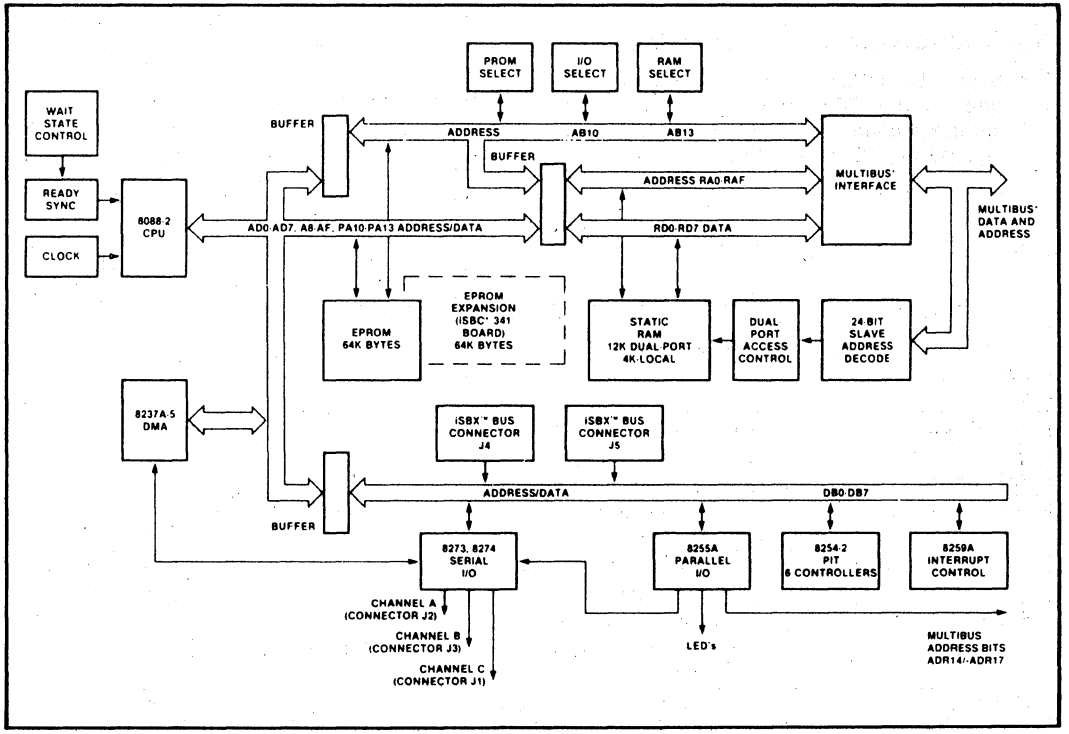


Figure 6. Block Diagram of the iSBC® 88/45 ADCP Board

Interrupt Request Generation

Listed in Table 4 are the devices and functions supported by interrupts on the iSBC 88/45 ADCP board. All interrupt signals are brought to the interrupt jumper matrix. Any of the 23 interrupt sources are strapped to the appropriate 8259A PIC request level. The PIC resolves requests according to the software selected mode and, if the interrupt is unmasked, issues an interrupt to the CPU.

EPROM/RAM Expansion

In addition to the on-board RAM, the iSBC 88/45 ADCP board provides four 28-pin JEDEC sockets for EPROM expansion. By using 2764 EPROMs, the board has 32K bytes of program storage. Three of the JEDEC standard sockets also support byte-wide static RAMs or iRAMs; using 8K x 8 static RAMs provides an additional 24K bytes of RAM.

Inserting the optional iSBC 341 MULTIMODULE EPROM expansion board onto the iSBC 88/45 ADCP board provides four additional 28-pin JEDEC sites. This expansion doubles the available program storage or extends the RAM capability by 32K bytes.

iSBX™ MULTIMODULE™ Expansion

Two 8-bit iSBX MULTIMODULE connectors are provided on the iSBC 88/45 microcomputer. Through these connectors, additional iSBX functions extend the I/O capability of the microcomputer. The iSBX connectors provide the necessary signals to interface to the local bus.

In addition to specialized or custom designed iSBX boards, the customer has a broad range of Intel iSBC MULTIMODULEs available, including parallel I/O, analog I/O, IEEE 488 GPIB, floppy disk, magnetic bubbles, video, and serial I/O boards.

The serial I/O MULTIMODULE boards include the iSBX 351 (one ASYNC/SYNC serial channel) the iSBX 352 (one HDLC/SDLC serial channel) and the iSBX 354 (two SYNC/ASYNC, HDLC/SDLC serial channels) boards. Adding two iSBX 352 MULTIMODULE boards to the iSBC 88/45 ADCP provides a total of five HDLC/SDLC channels.

MULTIBUS® Multimaster Capabilities

OVERVIEW

The MULTIBUS system is Intel's industry standard microcomputer bus structure. Both 8- and 16-bit single board computers are supported on the MULTIBUS structure with 24 address and 16 data lines. In addition to expanding functions contained on a single board computer (e.g., memory and digital I/O), the MULTIBUS structure allows very powerful distributed processing configurations with multiple processors, intelligent slaves, and peripheral boards.

Multimaster Capability

The iSBC 88/45 ADCP board provides full MULTIBUS arbitration control logic. This control

Table 4. Interrupt Request Sources

Device	Function	No. of Interrupts
MULTIBUS* Interface	Select 1 interrupt from MULTIBUS* resident peripherals or other CPU boards	8
8273 HDLC/SDLC Controller	Transmit buffer empty and receive buffer full	2
8274 HDLC/SDLC SYNC/ASYNC Controller	Software examines register for status of communication operation	1
8254-Timer	Counter 2 of both PIT devices	2
iSBX™ Connectors	Function determined by iSBX™ MULTIMODULE™ Board (2 interrupts per socket)	4
Bus Fail Safe Timer	Indicates MULTIBUS* addressed device has not responded to command within 4 msec	1
Power Line Clock	Source of 60 MHz signal from power supply	1
Bus Flag Interrupt	Flag interrupt in byte location 1000H signals board reset or data handling request	2
iSBC* 337 Board	Numeric Data Processor generated status information	1
8237A-5	Signals end of 8237 DMA operation	1

logic allows up to three iSBC 88/45 ADCP boards or other bus masters, including iSBC 286, iSBC 86 and iSBC 86 family boards to share the system bus using a serial (daisy chain) priority scheme. By using an external parallel priority decoder, the MULTIBUS system bus could be shared among sixteen masters.

The Intel standard MULTIBUS Interprocessor Protocol (MIP) software, implemented as the Intel iMMX 800 package for iRMX 86 and iRMX 88 Real-Time Executives, fully supports multiple 8-and 16-bit distributed processor functions. The software manages the message passing protocol between microprocessors.

System Development Capabilities

The application development cycle for an iSBC 88/45 ADCP board is reduced and simplified through the usage of several Intel tools. The tools include the Intellec Series Microcomputer Development System, the ICE-88 In-Circuit Emulator, the iSDM 86 debug monitor software, and the iRMX 86 and iRMX 88 run-time support packages.

The Intellec Series Microcomputer Development System offers a complete development environment for the iSBC 88/45 software. In addition to the operating system, assembler, utilities and application debugger features provided with the system, the user optionally can utilize higher-level languages like PL/M, PASCAL, and FORTRAN.

The ICE-88 In-Circuit Emulator provides a link between the Intellec system and the target iSBC 88/45-based system for code loading and execution. The ICE-88 package assists the developer with the debugging and system integrating processes.

Run-Time Building Blocks

Intel offers run-time foundation software to support applications which range from general purpose to high-performance solutions. The iRMX 88 Real-time Multitasking Executive provides a multitasking structure which includes task scheduling, task management, intertask communications, and interrupt servicing for high-performance applications. The highly configurable modules make the system tailoring job easier whether one uses the compact executive or the complete executive with its variety of peripheral devices supported.

The iRMX 86 Operating System provides a very rich set of features and options to support sophisticated applications solutions. In addition to supporting real-time requirements, the iRMX 86 Operating System has a powerful, but easy-to-use human interface. When added to the sophisticated I/O system, the iRMX 86 Operating System is readily extended to support assembler, PL/M, PASCAL, and FORTRAN software development environments. The modular building block software lends itself well to customized application solutions.

SPECIFICATIONS

Word Size

Instruction — 8, 16, 24, or 32 bits
Data — 8 or 16 bits

System Clock

8 MHz — ±0.1%

NOTE: Jumper selectable for 4 MHz operation with iSBC 337 Numeric Data Processor module or ICE-88 product.

Cycle Time

Basic Instruction Cycle at 8.00 MHz — 1.25 μsec, 250 nsec (assumes instruction in the queue)

NOTE: Basic instruction cycle is defined as the fastest instruction time (i.e., two clock cycles).

Memory Cycle Time

RAM — 500 nsec (no wait states)
EPROM — jumper selectable from 500 nsec to 625 nsec.

On-Board RAM* —

K Bytes	Hex Address Range
16 (total)	0000-3FFF
12 (dual-ported)	1000-3FFF

* Four iSBC 88/45 EPROM sockets support JEDEC 24/28-pin standard EPROMs and RAMs (3 sockets); iSBC 341 (4 sockets)

Environmental Characteristics

Temperature — 0-55°C, free moving air across the base board and MULTIMODULE board

Humidity — 90%, non-condensing

Physical Characteristics

Width — 30.48 cm (12.00 in)
Length — 17.15 cm (6.75 in)
Height — 1.50 cm (0.59 in)
Weight — 6.20 gm (22 oz)

Memory Capacity/Addressing

On-Board EPROM* —

Device	Total K Bytes	Hex Address Range
2716	8	FE000-FFFFF
2732A	16	FC000-FFFFF
2764	32	F8000-FFFFF
27128	64	F0000-FFFFF

With optional iSBC® 341 MULTIMODULE™ EPROM —

Device	Total K Bytes	Hex Address Range
2716	16	FC000-FFFFF
2732A	32	F8000-FFFFF
2764	64	F0000-FFFFF
27128	128	E0000-FFFFF

* Four iSBC 88/45 EPROM sockets support JEDEC 24/28-pin standard EPROMs and RAMs (static and iRAM, 3 sockets); iSBC 341 sockets also support EPROMs and RAMs.

Timer Input Frequency — 8.00 MHz ±0.1%

Interfaces

iSBX™ Bus — All signals TTL compatible

Serial RS232C Signals —

CTS	CLEAR TO SEND
DSR	DATA SET READY
DTE TXC	TRANSMIT CLOCK
DTR	DATA TERMINAL READY
FG	FRAME GROUND
RTS	REQUEST TO SEND
RXC	RECEIVE CLOCK
RXD	RECEIVE DATA
SG	SIGNAL GROUND
TXD	TRANSMIT DATA

Serial RS422A/449 Signals —

CS	CLEAR TO SEND
DM	DATA MODE
RC	RECEIVE COMMON
RD	RECEIVE DATA
RS	REQUEST TO SEND
RT	RECEIVE TIMING
SC	SEND COMMON
SD	SEND DATA
SG	SIGNAL GROUND
TR	TERMINAL READY
TT	TERMINAL TIMING

Electrical Characteristics

DC Power Dissipation — 28.3 Watts

DC Power Requirements —

Configuration	Current Requirements (all voltages ±5%) +5V	+12V	−12V
without EPROM[1]	5.1A	20 mA	20 mA
with 8K EPROM (using 2716)	+0.14A	—	—
with 16K EPROM (using 2732A)	+0.20A	—	—
with 32K EPROM (using 2764)	+0.24A	—	—
with 64K EPROM (using 27128)	+0.24A	—	—

NOTE 1: AS SHIPPED - no EPROMs in sockets, no iSBC 341 module. Configuration includes terminators for two RS422A/449 and one RS232C channels.

Serial Communication Characteristics

Channel	Device	Supported Interface	Max. Baud Rate
A	8274[1]	RS442A/449	800K SDLC/HDLC
		RS232C	125K Synchronous
		CCITT V.24	50K Asynchronous
B	8274	RS232C	125K Synchronous[2]
		CCITT V.24	50K Asynchronous
C	8273[3]	RS442A/449	64K SDLC/HDLC[3]
		RS232C	9.6K SELF CLOCKING
		CCITT V.24	

NOTES:
1. 8274 supports HDLC/SDLC/SYNC/ASYNC multiprotocol
2. Exceed RS232C/CCITT V.24 rating of 20K baud
3. 8273 supports HDLC/SDLC

BAUD RATE EXAMPLES (Hz)

8254 Timer Divide Count N	Synchronous K Baud	Asynchronous ÷16 K Baud	÷32 K Baud	÷64 K Baud
10	800	50.0	25.0	12.5
26	300	19.2	9.6	4.8
31	256	16.1	8.06	4.03
52	154	9.6	4.8	2.4
104	76.8	4.8	2.4	1.2
125	64	4.0	2.0	1.0
143	56	3.5	1.7	.87
167	48	3.0	1.5	.75
417	19.2	—	—	—
833	9.6	—	—	—
EQUATION	$\frac{8,000,000}{N}$	$\frac{500K}{N}$	$\frac{250K}{N}$	$\frac{125K}{N}$

SERIAL INTERFACE CONNECTORS

Interface	Mode[1]	MULTIMODULE™ Edge Connector	Cable	Connector
RS232C	DTE	26-pin[4], 3M-3462-0001	3M[2]-3349/25	25-pin[6], 3M-3482-1000
RS232C	DCE	26-pin[4], 3M-3462-0001	3M[2]-3349/25	25-pin[6], 3M-3483-1000
RS449	DTE	40-pin[5], 3M-3464-0001	3M[3]-3349/37	37-pin[7], 3M-3502-1000
RS449	DCE	40-pin[5], 3M-3464-0001	3M[3]-3349/37	37-pin[7], 3M-3503-1000

NOTES:
1. DTE — Data Terminal Equipment mode (male connector); DCE — Data Circuit Equipment mode (female connector) requires line swaps.
2. Cable is tapered at one end to fit the 3M-3462 connector.
3. Cable is tapered to fit 3M-3464 connector.
4. Pin 26 of the edge connector is not connected to the flat cable.
5. Pins 38, 39, and 40 of the edge connector are not connected to the flat cable.
6. May be used with the cable housing 3M-3485-1000.
7. Cable housing 3M-3485-4000 may be used with the connector.

Line Drivers (supplied)

Device	Characteristic	Qty	Installed
1488	RS232C	3	1
1489	RS232C	3	1
3486	RS422A	2	2
3487	RS422A	2	2

Reference Manual

143824 — iSBC 88/45 Advanced Data Communications Processor Board Hardware Reference Manual (not supplied).

Reference manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, CA 95051

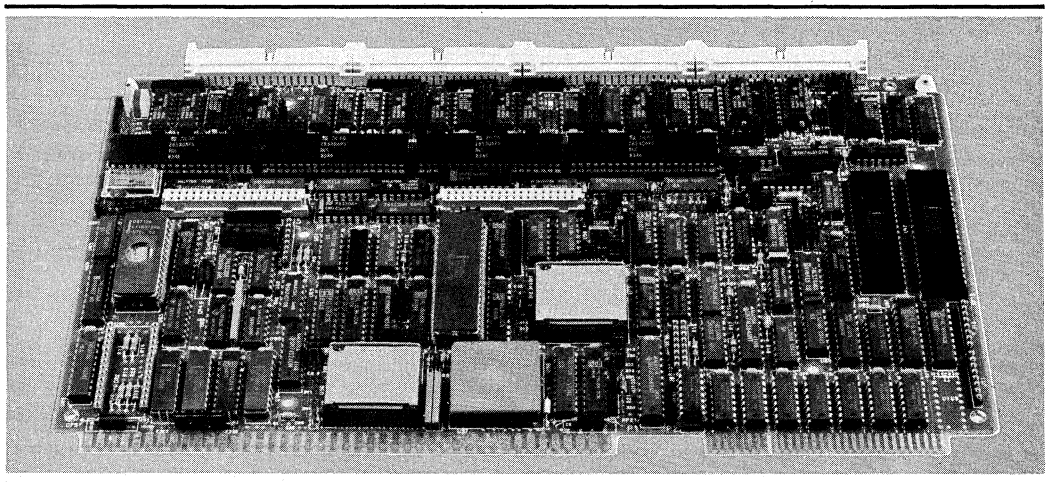
ORDERING INFORMATION

Part Number	Description
SBC 88/45	8-bit 8088-based Single Board Computer with 3 HDLC/SDLC serial channels

iSBC® 188/48 ADVANCED COMMUNICATING COMPUTER

- **iSBC® Single Board Computer or Intelligent Slave Communication board**
- **8 Serial Communications channels, expandable to 12 channels on a single MULTIBUS® board**
- **6 MHz iAPX 188 Microprocessor**
- **Supports RS232C interface on 6 channels, RS422A/449 or RS232C interface configurable on 2 channels**
- **Supports Async, Bisync HDLC/SDLC, on-chip baud rate generation, half/full-duplex, NRZ, NRZI or FM encoding/decoding**
- **7 on-board DMA channels for serial I/O, 2 80188 DMA channels for iSBX™ MULTIMODULE™ board**
- **MULTIBUS® Interface for system expansion and Multimaster configuration**
- **2 iSBX™ connectors for low cost I/O expansion**
- **64K Bytes Dual-ported RAM expandable to 192K Bytes with Parity using the iSBC® 307 RAM MULTIMODULE™ board**
- **2 28-pin JEDEC PROM sites expandable to 6 sites with the iSBC® 341 MULTIMODULE™ board for a maximum of 192K Bytes EPROM**
- **Resident firmware to handle up to 12 RS232C Async lines**
- **Optional Operating System firmware**

The iSBC® 188/48 Advanced Communicating Computer (COMMputer™) is an intelligent 8-channel single board computer. This iSBC board adds 6MHz iAPX 188 microprocessor-based communications flexibility to the Intel line of OEM microcomputer systems. Acting as a stand-alone CPU or intelligent slave for communication expansion, this board provides a high performance, low-cost solution for multi-user systems. The features of the iSBC 188/48 board are uniquely suited to manage higher-layer protocol requirements needed in today's data communications applications. This single board computer takes full advantage of Intel's VLSI technology to provide state-of-the-art, economic, computer-based solutions for OEM communications-oriented applications.



SEPTEMBER 1984
ORDER NUMBER: 230890-002

OPERATING ENVIRONMENT

The iSBC 188/48 COMMputer™ features have been designed to meet the needs of numerous communications applications. Typical applications include:

1. Terminal/cluster controller
2. Front-end processor
3. Stand-alone communicating computer

Terminal/cluster controller

A terminal/cluster controller concentrates communications in a central area of a system. Efficient handling of messages coming in or going out of the system requires sufficient buffer space to store messages and high speed I/O channels to transmit messages. More sophisticated applications, such as cluster controllers, also require character and format conversion capabilities to allow different types of terminals to be attached.

The iSBC 188/48 Advanced Communicating Computer is well suited for multi-terminal systems (See Figure 1). Up to 12 serial channels can be serviced in multi-user or cluster applications by adding two iSBX 354 MULTIMODULE boards. The dual-port RAM provides a large on-board buffer to handle incoming and outgoing messages at data rates up to 19.2K Baud. Two channels are supported for continuous data rates greater than 19.2K Baud. Each serial channel can be individually programmed for different Baud rates to allow system configurations with differing terminal types. The firmware supplied on the iSBC 188/48 board supports up to 12 asynchronous RS232C serial channels, provides modem control and performs power-up diagnostics. The high performance of the on-board CPU provides intelligence to handle protocols and character handling typically assigned to the system CPU. This distribution of intelligence results in optimizing system performance by releasing the system CPU of routine tasks.

Front-end Processor

A front-end processor off-loads a system's central processor of tasks such as data manipulation and text editing of characters collected from the attached terminals. A variety of terminals require flexible terminal interfaces. Program code

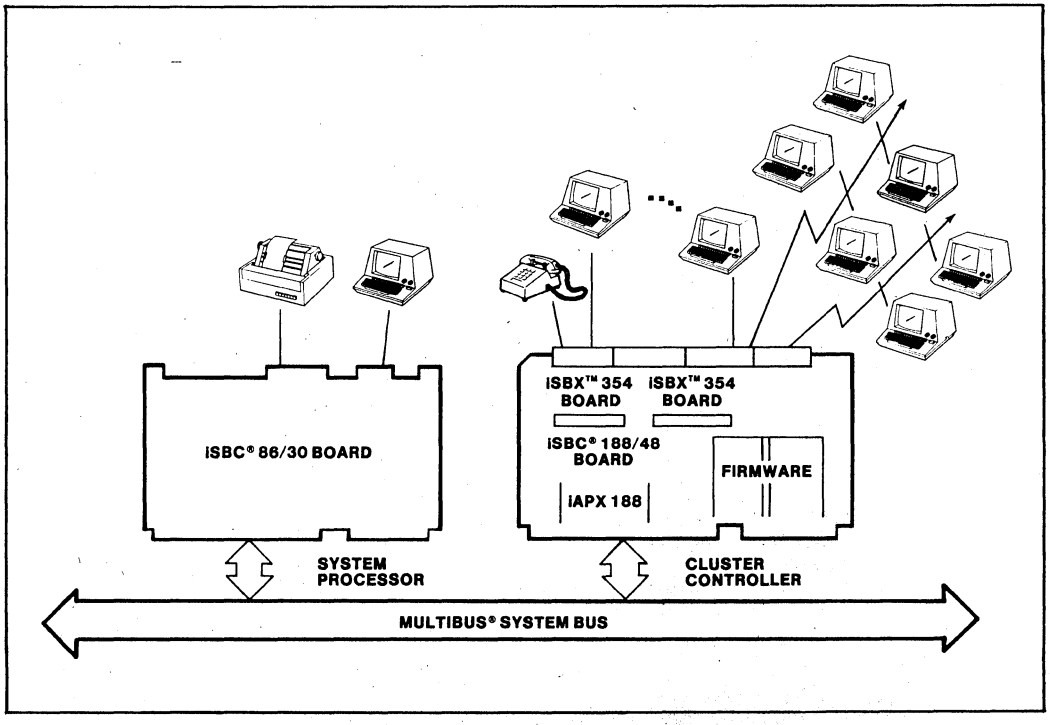


Figure 1. Terminal/Cluster Controller Application

is often dynamically down-loaded to the front-end processor from the system CPU. Downloading code requires sufficient memory space for protocol handling and program code. Flow control and efficient handling of interrupts require an efficient operating system to manage the hardware and software resources.

The iSBC 188/48 board features are designed to provide a high performance solution for front-end processor applications (see Figure 2). A large amount of random access memory is provided for dynamic storage of program code. In addition, local memory sites are available for storing routine programs such as X.25, SNA or bisync protocol software. The serial channels can be configured for links to mainframe systems, point-to-point terminals, modems or multi-drop configurations.

STAND-ALONE COMMputer™ APPLICATION

A stand-alone communicating computer is a complete computer system. The CPU is capable of managing the resources required to meet the needs of multi-terminal, multi-protocol applications. These applications typically require multi-terminal support, floppy disk control, local memory allocation, and program execution and storage.

To support stand-alone applications, the iSBC 188/48 COMMputer board can combine the computational capabilities of an on-board CPU with the nucleus of a real-time operating system (optional) to provide a high-speed system solution controlling 8 to 12 channels of serial I/O (see Figure 3). The local memory available is large enough to handle special purpose code, execution code and routine protocol software. The MULTIBUS interface can be used to access additional system functions. Floppy disk control and graphics capability can be added to the iSBC stand-alone computer through the iSBX connectors.

ARCHITECTURE

The four major functional areas are Serial I/O, CPU, Memory and DMA. These areas are illustrated in Figure 4.

Serial I/O

Eight HDLC/SDLC serial interfaces are provided on the iSBC 188/48 board. The serial interface can be expanded to 12 channels by adding 2

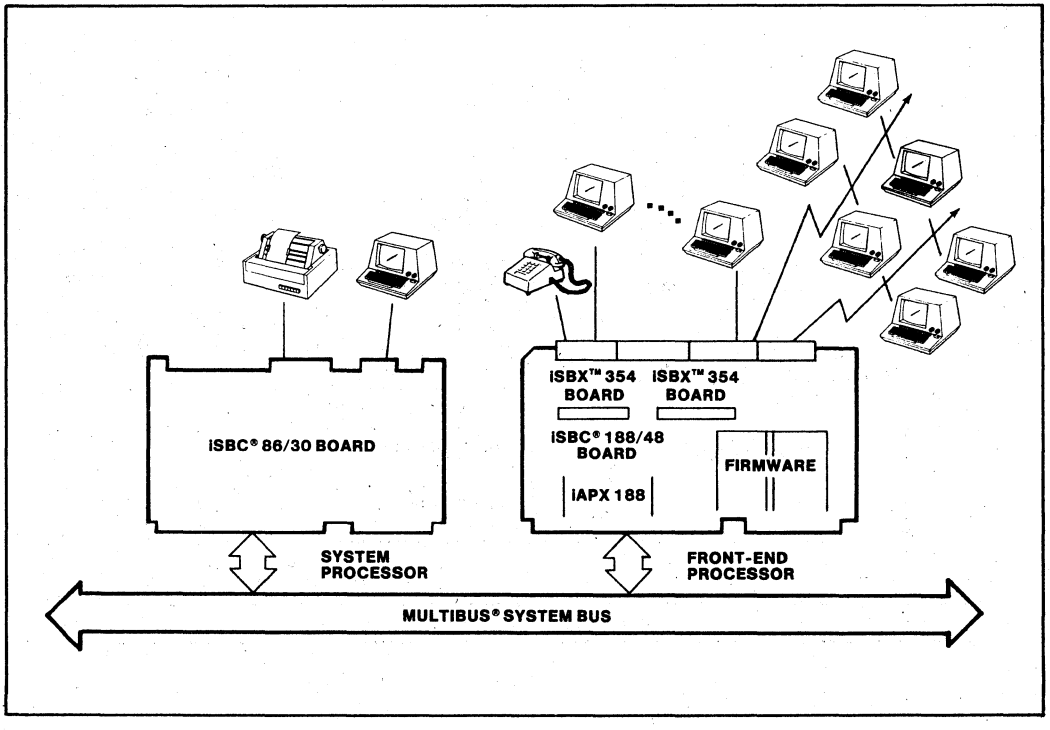


Figure 2. Front-end Processor Application

iSBX 354 MULTIMODULE boards. The HDLC/SDLC interface is compatible with IBM system and terminal equipment and with CCITT's X.25 packet switching interface.

Four 82530 Serial Communications Controllers (SCC) provide eight channels of half/full duplex serial I/O. Six channels support RS232C interfaces. Two channels are RS232C/422/449 configurable and can be tri-stated to allow multidrop networks. The 82530 component is designed to satisfy several serial communications requirements: asynchronous, byte-oriented synchronous, and bit-oriented synchronous (HDLC/SDLC) modes. The increased capability at the serial controller point results in off-loading the CPU of tasks formerly assigned to the CPU or its associated hardware. Configurability of the 82530 allows the user to configure it to handle all asynchronous data formats regardless of data size, number of start and stop bits, or parity requirements. An on-chip Baud rate generator allows independent Baud rates on each channel.

The clock can be generated either internally with the SCC chip, with an external clock or via the NRZ clock encoding mechanism.

All eight channels can be configured as Data Terminal Equipment (DTE) or Data Communication Equipment (DCE). Table 1 lists the interfaces supported.

Central CPU

The iAPX 188 central processor along with the optional Operating System Firmware component provides high performance, flexibility and powerful processing power. The 80188 component is a highly integrated microprocessor with an 8-bit data bus interface an a 16-bit internal architecture to give high performance. The iAPX 188 is upward compatible with iAPX 86 and iAPX 186 software. The O.S. component provides timers and interrupt controllers as well as the optional iRMX™ 86 nucleus primitives for those applications requiring a real-time executive.

The 80188/82530 combination with on-board PROM/EPROM sites, and dual-port RAM provide the intelligence and speed to manage multi-user, multi-protocol communications operations.

Memory

There are two areas of memory on-board: dual-port RAM and universal site memory. The iSBC

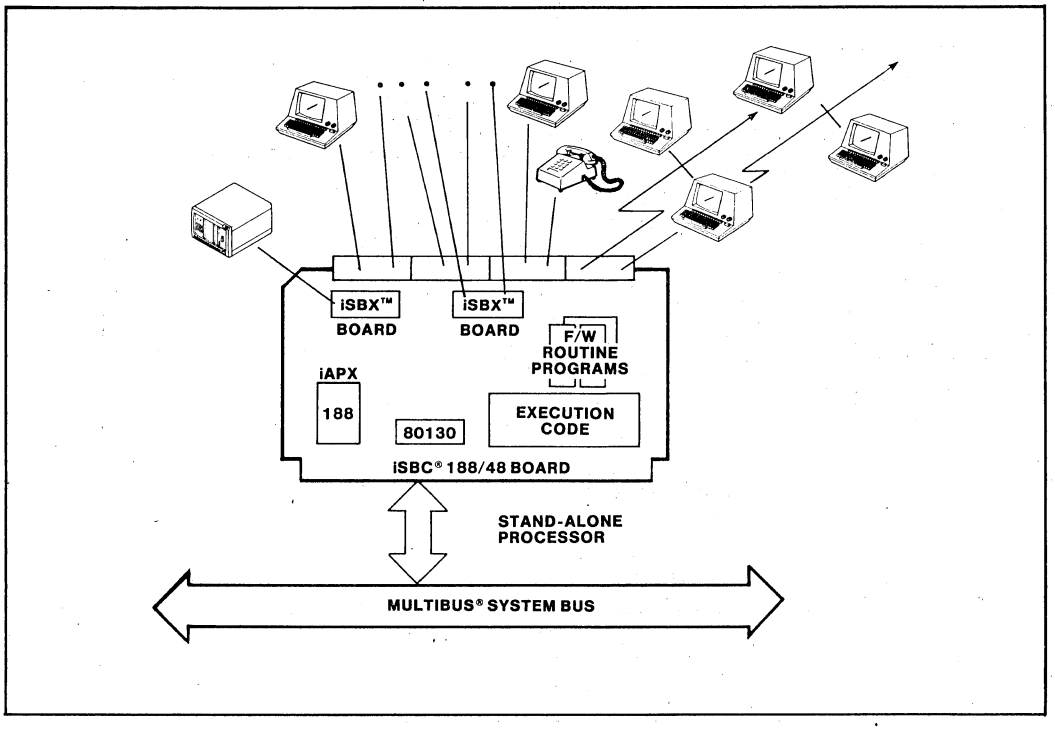


Figure 3. Stand-alone COMMputer™ Application

188/48 board contains 64K bytes of dual-port RAM that is addressable by the iAPX 188 on-board. The dual-port memory is configurable anywhere in a 16M Byte address space on 64K Byte boundaries as addressed from the MULTIBUS port. Not all of the 64K bytes are visible from the MULTIBUS side. The amount of dual-port memory visible to the MULTIBUS side can be set (with jumpers) to none, 16K bytes, or 48K bytes. The on-board RAM is expandable to a total of 192K bytes with parity by adding the iSBC 307 MULTIMODULE board. In a multiprocessor system these features provide local memory for each processor and shared system memory configurations where the total system memory size can exceed one megabyte without addressing conflicts.

The second area of memory is universal site memory providing flexible memory expansion. Two 28-pin JEDEC sockets are provided. One of these sockets is used for the resident firmware as described in the FIRMWARE section on Page 7.

The default configuration of the board supports 16K Byte EPROM devices such as the Intel 27128 component. However, these sockets can contain ROM, EPROM, Static RAM, or EEPROM. Both sockets must contain the same type of component (i.e. as the first socket contains an EPROM for the resident firmware, the second must also contain an EPROM with the same pinout). Up to 32K bytes can be addressed per socket giving a maximum universal site memory size of 64K bytes. By using the iSBC 341 MULTIMODULE board, a maximum of 192K bytes of universal site memory is available. This provides sufficient memory space for on-board network or resource management software.

Table 1. iSBC® 188/48 Interface Support

Connection	Synchronous Modem or Direct	Asynchronous Modem or Direct
Point-to-point	X**	X
Multidrop	Channels 0 and 1	Channels 0 and 1
Loop	X	N/A

** All 8 channels are denoted by X.

On-Board DMA

Seven channels of Direct Memory Access (DMA) are provided between serial I/O and on-board

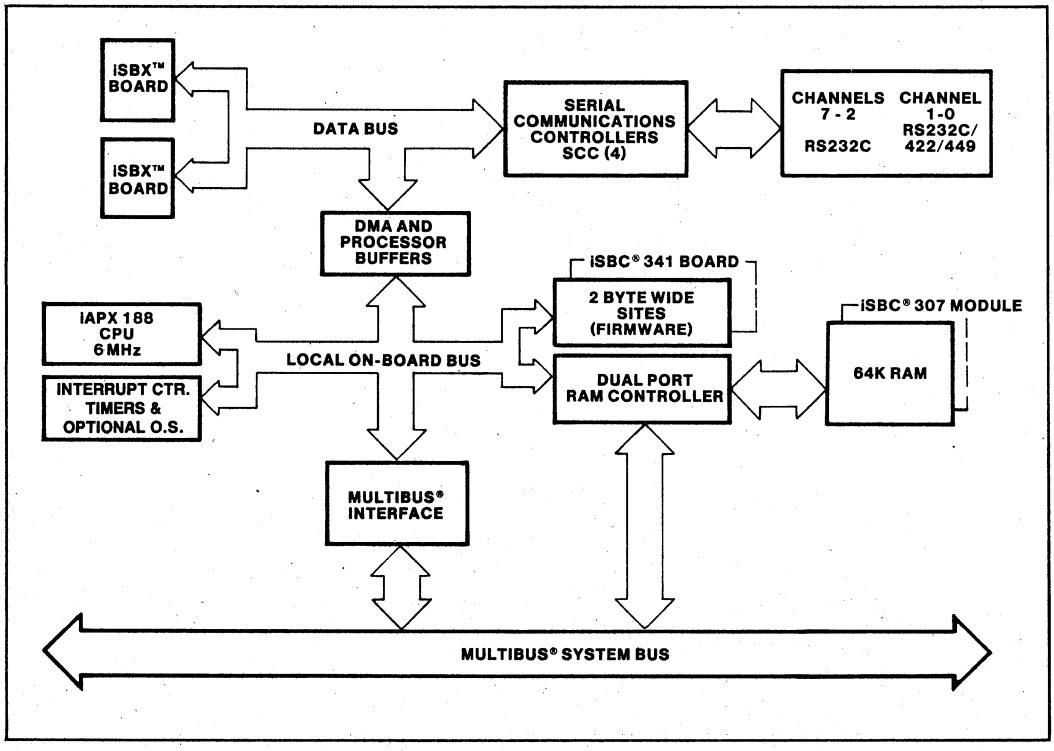


Figure 4. Block Diagram of iSBC 188/48 Board

dual-port RAM by two 8237-5 components. Each of channels 0,1, 2, 3, 5, 6, and 7 is supported by their own DMA line. Serial channels 0 and 1 are configurable for full duplex DMA. Configuring the full duplex DMA option for Channels 0 and 1 would require Channels 2 and 3 to be interrupt driven or polled. Channel 4 is interrupt driven or polled only.

Two DMA channels are integrated into the iAPX 188 processor. These additional channels can be connected to the iSBX interfaces to provide DMA capability to iSBX MULTIMODULE boards such as the iSBX 218A Floppy Disk Controller MULTIMODULE board.

OPERATING SYSTEM SUPPORT

Intel offers run-time foundation software to support applications that range from general purpose to high-performance solutions. Release 4 of the iRMX 88 Real-Time Executive provides an event-driven multitasking structure for the iSBC 188/48 board that includes task scheduling, task management, intertask communications and interrupt servicing for high-performance applications. Application tasks utilize intertask communications, asynchronous I/O control, priority-based resource allocation and file support for peripheral controllers. The small, high-performance iRMX 88 Executive can be located in EPROM or bootstrapped into iSBC 188/48 dual-port memory.

Release 6 of the iRMX 86 Operating System provides a rich set of features and options to support sophisticated stand-alone communications applications on the iSBC 188/48 Advanced Communicating Computer. In addition to supporting real-time requirements, the iRMX 86 Operating System Release 6 has a powerful, yet easy to use human interface. Services provided by the iRMX 86 Operating System include facilities for executing programs concurrently, sharing resources and information, servicing asynchronous events and interactively controlling system resources and utilities. The iRMX 86 Operating System is readily extended to support assembler, PL/M, PASCAL, and FORTRAN software development environments. The modular building block software lends itself well to customized application solutions. If the iSBC 188/48 is acting as an intelligent slave in a system environment, an iRMX 86 driver resident in the host CPU can be written by following the examples in Application Note 86, "Using the iRMX86 Operating System".

*UNIX is a trademark of Bell Laboratories.

SUPPORT FOR OPTIONAL OPERATING SYSTEM FIRMWARE

Release 3 of the iOSP 86 package provides the tools necessary to develope (P)ROM or RAM-based applications that use the optional Operating System Firmware component on the iSBC 188/48 board. All of the system initialization and run-time facilities are provided in libraries that may be configured to specific requirements and linked to application programs written in high level programming languages such as PL/M, Pascal or FORTRAN. The iOSP package also enables users to add higher level I/O functions from the fully compatible iRMX 86 Operating System, or to form custom, real-time systems. Please contact your local sales office for the implementation details of the operating system firmware.

The iSDM™ 86 System Debug Monitor supports target system debugging for the iSBC 188/48 Advanced Communicating COMMputer board. The monitor contains the necessary hardware, software and documentation required to interface the iSBC 188/48 target system to an Intel Microcomputer Development System for debugging application software.

The XENIX* 286 Operating System, Release 2, is a fully-licensed adaptation of the Bell Laboratories System III UNIX* Operating System. The XENIX system is an interactive, protected, multi-user, multi-tasking operating system with a powerful, flexible human interface. Release 2 of XENIX 286 includes a software driver for the iSBC 188/48 board (and up to two iSBX354 Multimodule Boards) acting as an intelligent slave for multi-user applications requiring multiple persons running independent, terminal-oriented jobs. Example applications include distributed data processing, business data processing, software development and engineering or scientific data analysis. XENIX 286 Release 2 Operating System services include device independent I/O, tree-structured file directory and task hierarchies, re-entrant/shared code and system accounting and security access protection.

FIRMWARE

The iSBC 188/48 Communicating COMMputer board is supplied with resident firmware that supports up to 12 RS232C asynchronous serial channels. In addition, the firmware provides a facility for a host CPU to download and execute code on the iSBC 188/48 board. Simple power-up confidence tests are also included to provide a quick diagnostic service. The firmware converts the iSBC 188/48 COMMputer to a slave communications controller. As a slave communications controller, it requires a separate MULTIBUS host CPU board and requires the use of a

Table 2. Features of the iSBC® 188/48 Firmware

Feature	Description
Asynchronous Serial Channel Support	Supports the serial channels in asynchronous ASCII mode. Parameters such as baud rate, parity generation, parity checking and character length can be programmed independently for each channel.
Block Data Transfer (On Output)	Relieves the host CPU of character-at-a-time interrupt processing. The iSBC 188/48 board accepts blocks of data for transmission and interrupts the processor only when the entire block is transmitted.
Limited Modem Control	Provides software control of the Data Terminal Ready (DTR) line on all channels. Transitions on the Carrier Detect (CD) line are sensed and reported to the host CPU.
Tandem Mode Support	Transmits an XOFF character when the number of characters in its receive buffer exceeds a threshold value and transmits an XON character when the buffer drains below some other threshold.
Download and execute capability	Provides a capability for the host CPU to load code anywhere in the address space of the iSBC 188/48 board and to start executing at any address in its address space.
Power Up Confidence Tests	On board reset, the firmware executes a series of simple tests to establish that crucial components on the board are functional.

MULTIBUS interrupt line to signal the host processor. Table 2 summarizes the features of the firmware.

INTERRUPT CAPABILITY

The iSBC 188/48 board has two programmable interrupt controllers (PICs). One is integrated into the 80188 processor and the other in the 80130 component. The two controllers are configured with the 80130 controller as the master and the 80188 controller as the slave. Two of the 80130 interrupt inputs are connected to the 82530 serial controller components to provide vector interrupt capablities by the serial controllers. The iSBC 188/48 board provides 22 interrupt levels. The highest level is the NMI (Non-Maskable Interrupt) line which is directly tied to the 80188 CPU. This interrupt is typically used for signaling catastrophic events (e.g. power failure). There are 5 levels of interrupts internal to the 80188 processor. Another 8 levels of interrupts are available from the 80130 component. Of these 8, one is tied to the programmable interrupt controller (PIC) of the 80188 CPU. An additional 8 levels of interrupts are available at the MULTIBUS interface. The iSBC 188/48 board does not support bus vectored interrupts. Table 3 lists the possible interrupt sources.

EXPANSION

EPROM/RAM Expansion

Memory may be expanded by adding Intel compatible memory expansion boards. The universal site memory can be expanded to six sockets by adding the iSBC 341 MULTIMODULE board for a maximum total of 192K bytes of universal site memory. The 64K bytes of on-board dual-port RAM can be expanded to a maximum total of 192K bytes by adding the iSBC 307 MULTIMODULE board. The iSBC 307 MULTIMODULE board also provides parity for all 192K bytes of on-board RAM.

iSBX™ MULTIMODULE™ Expansion

Two 8-bit iSBX MULTIMODULE connectors are provided on the iSBC 188/48 board. Using iSBX modules additional functions can be added to extend the I/O capability of the board. In addition to specialized or custom designed iSBX boards, there is a broad range of iSBX MULTIMODULE boards from Intel including parallel I/O, analog I/O, IEEE 488 GPIB, floppy disk, magnetic bubbles, video and serial I/O boards.

The serial I/O MULTIMODULE boards available include the iSBX 354 Dual Channel Expansion

MULTIMODULE board. Each iSBX 354 MULTIMODULE board adds two channels of serial I/O to the iSBC 188/48 board for a maxmimum of twelve serial channels. The 82530 serial communications controller on the MULTIMODULE handles a large variety of serial communications protocols. This is the same serial controller as is used on the iSBC 188/48 board to offer directly compatible expansion capability for the iSBC 188/48 COMMputer board.

MULTIBUS® INTERFACE

The iSBC 188/48 Advanced COMMputer board can be a MULTIBUS master or intelligent slave in a multimaster system. The iSBC 188/48 board incorporates a flag byte signalling mechanism for use in multiprocessor environments where the iSBC 188/48 board is acting as an intelligent slave. This mechanism provides an interrupt handshake from the MULTIBUS System Bus to the on-board processor and vice-versa.

The Multimaster capabilities of the iSBC 188/48 board offers easy expansion of processing capacity and the benefits of multiprocessing. Memory and I/O capacity may be expanded and additional functions added using Intel MULTIBUS compatible expansion boards.

Table 3. Interrupt Request Sources

Device	Function	Number of Interrupts
MULTIBUS® Interface INT0 - INT7	Requests from MULTIBUS resident peripherals or other CPU boards.	8
82530 Serial Controllers	Transmit buffer empty, receive buffer full and channel errors 1 and external status	8 per 82530 Total = 32
Internal 80188 Timer and DMA	Timer 0,1,2 outputs and 2 DMA channel interrupts	5
80130 Timer Outputs	Timer 0,1,2, outputs of 80130	3
Interrupt from Flag Byte Logic	Flag byte interrupt set by MULTIBUS master (through MULTIBUS® I/O Write)	1
Bus Flag Interrupt	Interrupt to MULTIBUS® (Selectable for INT0 to INT7) generated from on-board 80188 I/O Write	1
iSBX™ connectors iSBX™ DMA	Function determined by iSBX™ MULTIMODULE™ board DMA interrupt from iSBX™(TDMA)	4 (Two per connector) 2
Bus fail-safe timeout Interrupt	Indicates iSBC® 188/48 board timed out either waiting for MULTIBUS® access or timed out from no acknowledge while on MULTIBUS System Bus	1
Latched Interrupt	Converts pulsed event to a level interrupt. Example: 8237A-5 EOP	1
OR-gate Matrix	Concentrates up to 4 interrupts to 1 interrupt (selectable by stake pins)	1
Ring Indicator Interrupt	Latches a ring indicator event from serial channels 4,5,6, or 7	1
NOR-Gate Matrix	Inverts up to 2 interrupts into 1 (selectable by stake pins)	1

SPECIFICATIONS

Word Size

Instruction — 8, 16, 24 or 32 bits
Data Path — 8 bits

Processor Clock	82530 Clock	DMA Clock
6 MHZ	4.9152 MHz	3 MHz

MEMORY CAPACITY/ADDRESSING

Dual-Port RAM

iSBC®188/48 Board — 64K bytes

As viewed from the iAPX 188 — 64K

As viewed from the MULTIBUS® System Bus —
Choice: 0, 16K or 48K

EPROM

Using:

iSBC® 188/48 Board	Size	On Board Capacity	Address Range
2732	4K	8K	FE000-FFFFF$_H$
2764	8K	16K	FC000-FFFFF$_H$
27128	16K	32K	F8000-FFFFF$_H$
27256	32K	64K	F0000-FFFFF$_H$

Memory Expansion

1. **Ram Memory** — with iSBC 307 Board

   Total Capacity — 192K

   As viewed from the MULTIBUS® System Bus —
   Choice: 0, 16K or 48K Public
   16K to 192K Private
   64K or 192K Total

2. **EPROM with iSBC® board using:**

	Total Capacity	Address Range
2732	24K	F8000-FFFFF$_H$
2764	48K	F0000-FFFFF$_H$
27128	96K	E0000-FFFFF$_H$
27256	192K	C0000-FFFFF$_H$

I/O Capacity

Serial — 8 programmable lines using 4 82530 components
iSBX™ MULTIMODULE™ Board — 2 iSBX™ single-wide boards

Serial Communications Characteristics

Synchronous — Internal or external character synchronization on one or two synchronous characters

Asynchronous — 5-8 bits and 1, 1½ or 2 stop bits per character; programmable clock factor; break detection and generation; parity, overrun, and framing error detection

Baud Rates

Synchronous X1 Clock	
Baud Rate	**82530 Count Value (Decimal)**
64000	36
48000	49
19200	126
9600	254
4800	510
2400	1022
1800	1363
1200	2046
300	8190
Asynchronous X16 Clock	
Baud Rate	**82530 Count Value (Decimal)**
19200	6
9600	14
4800	30
2400	62
1800	83
1200	126
300	510
110	1394

INTERFACES

iSBX™ Bus

The iSBC 188/48 board meets iSBX compliance level D8/8 DMA

MULTIBUS® System Bus

The iSBC 188/48 board meets MULTIBUS compliance level Master/Slave D8 M24 I16 V0 EL

Serial RS232C Signals

CD	Carrier Detect
CTS	Clear to Send
DSR	Data Set Ready
DTE TXC	Transmit Clock
DTR	Data Terminal Ready
RTS	Request to Send
RXC	Receive Clock
RXD	Receive Data
SG	Signal Ground
TXD	Transmit Data
RI	Ring Indicator

RS422A/449 Signals

RC	Receive Common
RD	Receive Data
RT	Receive Timing
SD	Send Data
TT	Terminal Timing

ENVIRONMENTAL CHARACTERISTICS

Temperature — 0 to 55°C, at 200 Linear Feet/Min. (LFM) Air Velocity

Humidity — to 90%, non-condensing (25°C to 70°C)

PHYSICAL CHARACTERISTICS

Width: 30.48 cm (12.00 in)

Length: 17.15 cm (6.75 in)

Height: 2.90 cm (1.14 in)

Weight: 595 gm (21 ounces)

ELECTRICAL CHARACTERISTICS

The power required per voltage for the iSBC 188/48 board is shown below. These numbers do not include the current required by universal memory sites or expansion modules.

Voltage (Volts)	Current (Amps) typ.	Power (Watts) typ.
+ 5	4.56A	22.8W
+12	.12A	1.5W
−12	.11A	1.3W

ORDERING INFORMATION

Part Number	Description
iSBC 188/48	8-Serial Channel Advanced Communicating Computer

REFERENCE MANUAL

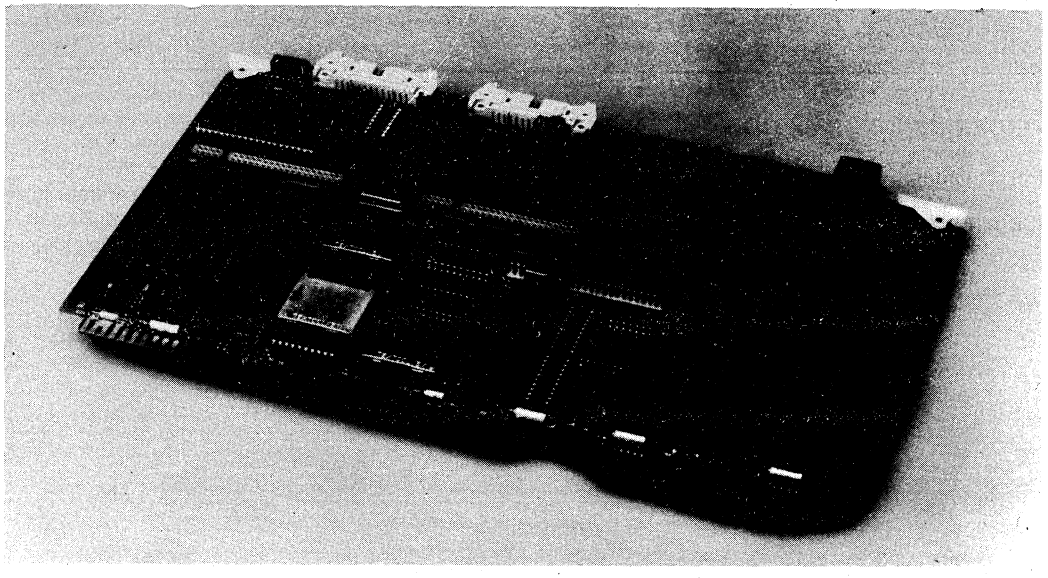
iSBC 188/48 Advanced Communications Computer Reference Manual
Order Number 146218-002
(146218-002 Available in Nov '84)

PRELIMINARY

iSBC® 186/51
COMMUNICATING COMPUTER

- 8 MHz iAPX 186 Microprocessor
- 128K Bytes of dual-ported RAM expandable on-board to 256K Bytes
- 82586 Local Communications Controller for CSMA/CD applications and 82501 Ethernet serial interface for Ethernet/ IEEE 802 specifications
- Two serial interfaces, RS-232C and RS-422A/RS-449 compatible
- Sockets for up to 192K Bytes of JEDEC 28 pin standard memory devices
- 80130 Real-Time Operating System Firmware
- Two iSBX™ bus connectors
- 16M Bytes address range of MULTIBUS®
- MULTIBUS® interface for multimaster configurations and system expansion
- Supported by a complete family of single board computers, peripheral controllers, digital & analog I/O, memory, packaging and software

The iSBC® 186/51 COMMUNICATING COMPUTER is a member of Intel's large COMMputer™ family of microcomputer products that utilizes Intel's VLSI technology to provide an economical self-contained computer for applications in data communications and local area network control. The combination of the iAPX 186 Central Processing Unit/80130 Operating System Firmware and the 82586 Local Communications Controller/82501 Ethernet Serial Interface makes it ideal for applications which require both communication and processing capabilities such as networked workstations, factory automation, office automation, communications servers, and many others. The CPU, Ethernet interface, serial communications interface, 128K Bytes of RAM, up to 192K Bytes of ROM, Operating System Firmware, I/O ports and drivers and the MULTIBUS® interface all reside on a single 6.75″ x 12.00″ printed circuit board.



JULY 1983
ORDER NUMBER: 230718-001

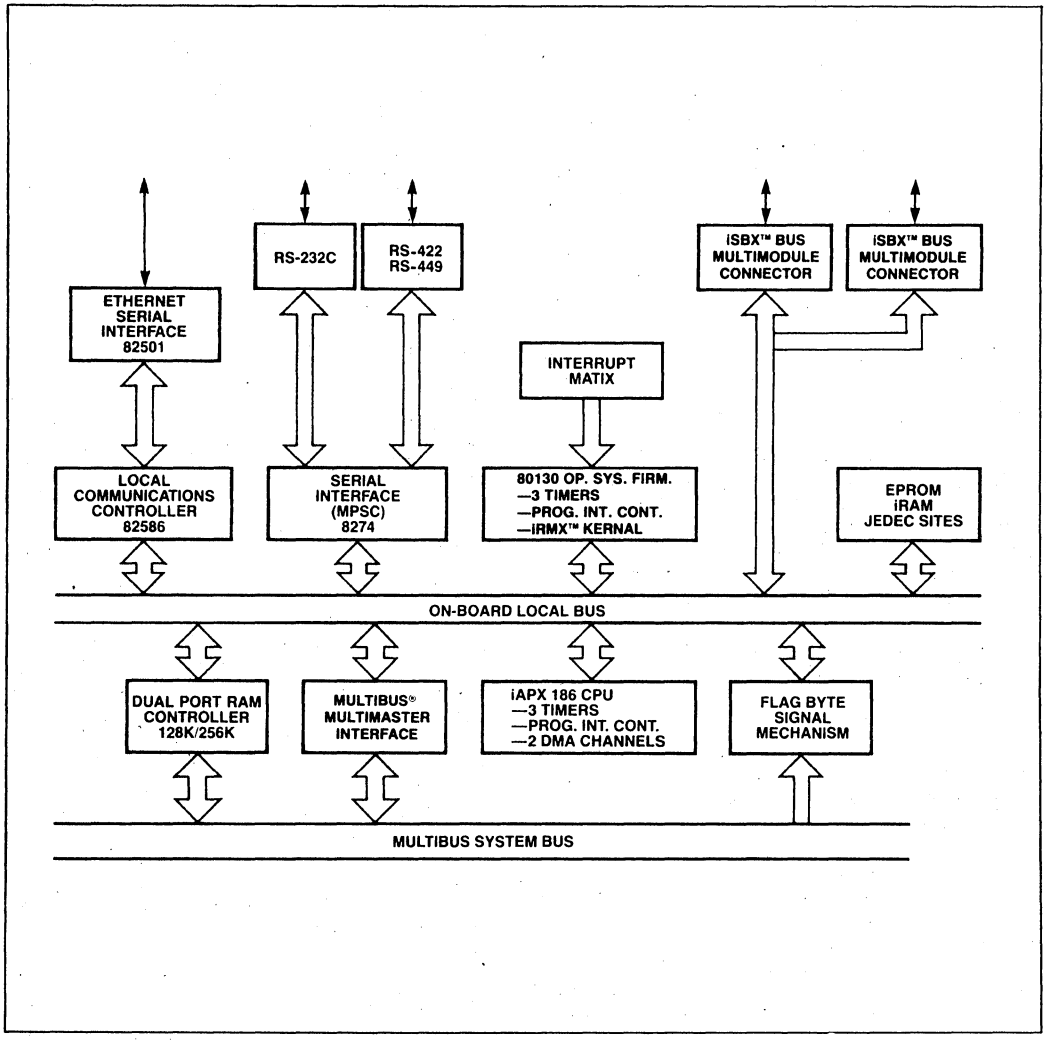


Figure 1. iSBC® 186/51 Block Diagram

FUNCTIONAL DESCRIPTION

Communicating Computer

The iSBC® 186/51 board integrates a programmable processor and communications capability onto one board, serving both computational and networking capacities as dictated by the application. The communications co-processor (82586) aids in this task by accomplishing as much of the communications task as possible before the processor intervenes (thus reducing the overhead load of the 80186 processor).

The integration of the communication and processing capabilities onto one board results in two primary benefits: (1) increased performance with elimination of system bus arbitration and (2) increased savings due to the compact, one-board design.

The dual capabilities of the iSBC 186/51 are useful in three types of applications: (1)as a single board communicating computer running both user applications and communications tasks; (2)as one bus master of a multiple processor board solution running a portion of the overall user application and the communications tasks; and (3) as an "intelligent bus slave" that performs communications related tasks as a peripheral processor to one or more bus masters in a communications intensive environment.

Architecture

The iSBC 186/51 board is functionally partitioned into three major sections: central computer, I/O including LAN interconnect and memory including shared dual port RAM (Figure 1).

The central computer, with an iAPX 186 CPU and the 80130 Operating System Firmware (OSF) provides powerful processing capability. The microprocessor and OSF primitives, together with the on-board PROM/EPROM sites, programmable timers/counters, and programmable interrupt control provide the intelligence to manage sophisticated communications operations on-board the iSBC 186/51. The timers/counters and interrupt control are also common to the I/O area providing programmable baud rates to the USARTs and prioritizing interrupts generated from the USARTs. The central computer functions are protected for access by the on-board 80186 only.

The I/O is centered around the Ethernet access provided by the 82586/82501 pair. All CSMA/CD protocols can be supported. Included here as well are two serial interfaces, both of which are fully programmable. In support of the single board computer, two iSBX connectors are provided for further customer expansion of I/O capabilities. The I/O is under full control of the on-board CPU and is protected from access by other system bus masters.

The third major segment, dual-port RAM memory, is the key link between the 80186, the Ethernet controller, and bus masters (if any) managing the system functions. The dual-port concept allows a common block of dynamic memory to be accessed by the on-board 80186 CPU, the on-board Ethernet controller and off-board bus masters. The system program can, therefore, utilize the shared dual-port RAM to pass command and status information between the bus masters and on-board CPU and Ethernet controllers. In addition, the dual-port concept permits blocks of data transmitted or received to accummulate in the on-board shared RAM, minimizing the need for a dedicated memory board.

CENTRAL COMPUTER FUNCTIONALITY

Central Processing Unit

The central processor for the iSBC 186/51 is Intel's iAPX 186 CPU. The iAPX 186 is a high integration 16-bit microprocessor. It combines several of the most common system components onto the chip (i.e., Direct Memory Access, Interval Timers, Clock generator, and Programmable Interrupt Controller) and provides a performance improvement of 30% over the 8086-2 processor. The CPU architecture includes four 16-bit Byte addressable data registers, two 16-bit index registers and two 16-bit memory base pointer registers. These are accessible by a total of 24 operand addressing modes for (1) comprehensive memory addressing, and (2) support of the data structures required for today's structured, high level languages—as well as assembly language.

Instruction Set

The iAPX 186 instruction set is a superset of the 8086. It maintains object code compatibility while adding 10 new instructions to the existing iAPX 86 instruction set. The iAPX 186 retains the variable length instruction format (including double operand instructions), 8-bit and 16-bit signed and unsigned arithmetic operators for binary, BCD and unpacked ASCII data, and iterative word and byte string manipulations. Added instructions include: Block I/O, Enter and Leave subroutines, Push Immediate, Multiply Quick, Array Bounds Checking, Shift and Rotate by Immediate, and Pop and Push All.

Architectural Features

A six-byte instruction queue provides prefetching of sequential instructions and can reduce the 750 nsec minimum instruction cycle to 250 nsec for queued instructions. The stack oriented architecture readily supports modular programming by facilitating fast, simple, intermodule communication, and other programming constructs needed for asynchronous real-time systems. Using a windowing technique and external logic, the full 16M Bytes addressing range of the IEEE-796 MULTIBUS Standard is available to the user. The dynamic relocation scheme allows ease in segmentation of pure procedure and data for efficient memory utilization. Four segment registers (code, stack, data, extra) contain program loaded offset values which are used to map 16-bit addresses to 20-bit addresses. Each register maps 64K Bytes at a time and activation of a specific register is controlled, both explicitly by program control, and implicitly by specific functions and instructions. A flag byte signaling mechanism aids in creating an interprocessor communication scheme. This includes (1) the ability to set/reset interrupts with MULTIBUS commands and (2) board reset.

OPERATING SYSTEM FUNCTIONALITY

Operating System Firmware

The 80130 provides a set of multitasking kernel primitives, kernel control storage, and the additional support hardware, including system timers and interrupt controller, required by those primitives. To the applications programmer, the OSF extends the iAPX 186 architecture by providing 35 operating system primitive instructions, and supporting five new system data types. This makes the OSF a logical and easy to use architectural extension to the iAPX 186 system design. The chip has also been designed to be compatible with the iRMX86 operating system.

Architecture

The 80130 is connected directly to the local bus of the 80186 processor with address decoding, buffering, and bus-demultiplexing logic contained on-chip (Figure 1). Internally, the 80130 firmware consists of two sections: an operating system unit and a control unit. The former consists of a 16K Byte operating-system-kernel control store complete with an operating-system timer, a delay timer, a bit-rate generator, and 8259A-compatible programmable interrupt logic.

The first timer generates the fundamental real-time clock period in the system. It is set to 10 milliseconds initially but can be modified by the system designer. The delay timer supports the kernel timing function by indicating the next event. Both these timer resources are reserved for use by the kernel.

The bit-rate generator, which has a range of 75 to 768 kilobits per second, is provided as a user resource. The 80130 interrupt logic vectors eight independent priority levels, one of which is reserved for the operating-system timers.

Operation

The 80130 supplements the 80186's basic architecture with five new objects, or system data types: jobs, tasks, segments, mailboxes, and regions. See Tables 1 and 2 for the new data types and operating system primitives.

The 80130 operates by creating, manipulating and deleting individual system objects. When an object is created, the 80130 returns its name to the creating task. This name is referred to and used as an abstract data type, called a TOKEN. The TOKEN is a highly efficient way of accessing the iSBC 186/51 address space. Referring to a segment object, for example, causes a 16-bit address to be loaded into one of the processor segment registers, which can then be used to directly address a paragraph (16-Byte unit) anywhere in the 1M Byte address space. Task creation is also accomplished in this manner and requires only the specification of a priority, a task private data segment (if needed), a task stack, and a task program starting address.

To take full advantage of multiprogramming, the operating system must provide each application with a separate environment—that is, separate memory and tasks. This isolation both protects independent programs from interfering with one another and allows the application programmer to work without regard to the other application programs in the system. The 80130 supports multiprogramming with the job data type. The creation of a job requires the specification of a large number of parameters and is normally done only when the system is being initialized.

Table 1. System Data Types Used in 80130 Operating System Firmware

Job	Jobs are the means of organizing the program environment and resources. An application consists of one or more jobs. Each iAPX 186 system data type is contained in some job. Jobs are independent of each other, but they may share access to resources. Each job has one or more tasks, one of which is an initial task. Jobs are given pools of memory, and they may create subordinate offspring jobs, which may borrow memory from their parents.
Task	Tasks are the means by which computations are accomplished. A task is an instruction stream with its own execution stack and private data. Each task is part of a job and is restricted to the resources provided by its job. Tasks may perform general interrupt handling as well as other computational functions. Each task has a set of attributes, maintained for it by the iAPX 186, which characterize its status. These attributes are: its containing job its register context its priority (0-255) its execution state (asleep, suspended, ready, running, asleep/suspended) its suspension depth its user-selected exception handler its option 8087 extended task state
Segment	Segments are the units of memory allocation. A segment is a physically contiguous sequence of 16-Byte, 8086 paragraph-length, units. Segments are created dynamically from the free memory space of a Job as one of its Tasks requests memory for its use. A segment is deleted when it is no longer needed. The iAPX 186 maintains and manages free memory in an orderly fashion, it obtains memory space from the pool assigned to the containing job of the requesting task and returns the space to the job memory pool (or the parent job pool) when it is no longer needed. It does not allocate memory to create a segment if sufficient free memory is not available to it; in that case it returns an error exception code.
Mailbox	Mailboxes are the means for intertask communication. Mailboxes are used by tasks to send and receive message segments. The iAPX 186 creates and manages two queues for each mailbox. One of these queues contains message segments sent to the mailbox but not yet received by any task. The other mailbox queue consists of tasks that are waiting to receive messages. The iAPX 186 assures that waiting tasks receive messages as soon as messages are available. Thus at any moment one or possibly both of two mailbox queues will be empty.
Region	Regions are the means of serialization and mutual exclusion. Regions are familiar as "critical code regions." The iAPX 186 region data type consists of a queue of tasks. Each task waits to execute in mutually exclusive code or to access a shared data region, for example to update a file record.
Tokens	The OSP interface makes use of a 16-bit TOKEN data type to identify individual OSF data structures. Each of these (each instance) has its own unique TOKEN. When a primitive is called, it is passed the TOKENs of the data structures on which it will operate.

Table 2. 80130 Operating System Firmware Primitives

JOB	CREATE JOB	Creates a job partition including memory pool, task list, and stack area.
TASK	CREATE TASK	Creates a task with the specified environment and priority and puts it in the ready state. Checks for insufficient memory available within the containing job.
	DELETE TASK	Deletes a task from the system as well as from any queues in which it is waiting. The task's state and stack segment are de-allocated.
	SUSPEND TASK	Suspends a task (changes its status to suspended) or increases the task's suspension count by 1. A sleeping task may also be suspended and will then awaken suspended unless resumed.
	RESUME TASK	Decreases the suspension count of a task by 1. If the count is at that point reduced to 0, the task state is made ready or if it was suspend-asleep, it is put back to asleep.
	SLEEP	Puts the task in the asleep state, a number of 10-ms units may be specified.
INTERRUPT	SET PRIORITY	Changes the task's priority to the value passed in the primitive.
	SET INTERRUPT	Assigns an interrupt handler to a level. The task that makes this call is made the interrupt task for the same level, unless the call indicates there is no interrupt task.
	RESET INTERRUPT	Disables an interrupt level. Cancels the interrupt handler, deletes the interrupt task for that level if assigned.
	GET LEVEL	Returns the number of the interrupt level for highest priority interrupt handler currently in operation (several interrupt handlers could be operating).
	EXIT INTERRUPT	Completes interrupt processing and sends end-of-interrupt signal to hardware.
	SIGNAL INTERRUPT	Invokes the interrupt task assigned to a level from that level's interrupt handler.
	WAIT INTERRUPT	Makes the interrupt task state suspended pending a signal interrupt from an interrupt handler. Used by an interrupt task to signal its readiness to service an interrupt.
	ENABLE	Enables an external interrupt level.
	DISABLE	Disables an external interrupt level.
	GET EXCEPTION HANDLER	Reads the location and exception-handling mode of the current operating system exception handler for a task.
	SET EXCEPTION HANDLER	Establishes the location and exception-handling mode of the current operating system exception handler for a task.

Table 2. 80130 Operating System Firmware Primitives (Cont.)

SEGMENT	CREATE SEGMENT	Allocates dynamically an area of memory of a specified length in 16-Byte paragraph units up to a maximum of 64K Bytes (for example, for use as a buffer). Returns a location token for the segment allocated.
	DELETE SEGMENT	De-allocates the memory segment indicated by the parameter token.
	ENABLE DELETION	Allows the system data type value indicated by the location token to be deleted.
	DISABLE DELETION	Prevents the system data type value indicated by the location token from being deleted.
MAILBOX	CREATE MAILBOX	Creates a mailbox with the specified task queueing discipline. Returns a location token.
	DELETE MAILBOX	Deletes a mailbox, and returns its memory. If tasks are waiting for the mailbox, they are awakened (their state is made ready) with an appropriate exception condition. If messages are waiting for tasks, they are discarded.
	SEND MESSAGE	Sends a message segment to a mailbox.
	RECEIVE MESSAGE	A task is ready to receive a message at a mailbox. The task is placed on the mailbox task queue. The task may optionally wait for a response indefinitely, or a number of time intervals (generally 10 ms long), or not at all. When complete, the primitive returns to the task the location token of the message segment received.
REGION	CREATE REGION	Creates a region data type value specifying a queueing discipline. Returns a token for the region.
	DELETE REGION	Deletes a region if and only if the region is not in use.
	ACCEPT CONTROL	Gains control of a region if it is immediately available, but does not wait if it is not available.
	RECEIVE CONTROL	Is the same primitive as accept control but the task that performs it may elect to wait.
	SEND CONTROL	Relinquishes a region.

Programmable Timers

The 80186 provides three internal 16-bit programmable timers. Two of these are highly flexible and are connected to four external pins (two per timer). They can be used to count external events, time external events, generate nonrepetitive waveforms, etc. The third timer is not connected to any external pins, and is useful for real-time coding and time delay applications. In addition, this third timer can be used as a prescaler to the other two, or as a DMA request source. The factory default configuration for timer 0 is baud rate generator.

The 80130 provides three more programmable timers. One is a factory default baud rate generator and outputs an 8254 compatible square wave to the RS232 Channel B. The other two timers are assigned to the use of the OSF and should not be altered by the user.

The system software configures each timer independently to select the desired function. Examples of available functions are shown in Table 3. The contents of each counter may be read at any time during system operation.

Interrupt Capability

The iSBC 186/51 has two programmable interrupt controllers (PICs): one in the 80186 component and one in the 80130 component. In the iRMX mode, the 80186 interrupt controller acts as a slave to the 80130. The 80186 interrupt controller in this mode uses all of its external interrupt pins. It therefore services only internally generated interrupts (i.e., three timers, two DMA channels). The 80130 interrupt controller operates in the master mode and has eight prioritized inputs that can be programmed either edge or level sensitive.

The iSBC 186/51 board provides 9 vectored interrupt levels. The highest level is the NMI (Non-Maskable Interrupt) line which is directly tied to the 80186 CPU. This interrupt is typically used for signaling catastrophic events (e.g., power failure). The Programmable Interrupt Controllers (PIC) provide control and vectoring for the next eight interrupt levels. As shown in Table 4, a selection of four priority processing modes is available for use in designing request processing configurations to match system requirements for efficient interrupt servicing with minimal latencies. Operating modes and priority assignments may be reconfigured dynamically via software at any time during system operation. The PIC accepts interrupt requests from all on-board I/O resources and from the MULTIBUS system bus. The PIC then resolves requests according to the selected mode and, if appropriate, issues an interrupt to the CPU.

Interrupt Request Generation

iSBC 186/51 Interrupt Service requests may originate from 25 sources. Table 5 contains a list of devices and functions supported by interrupts. All interrupts are jumper configurable with either suitcase or wire wrap to the desired interrupt request level.

Table 3. 80186 Programmable Timer Functions

Function	Operation
Interrupt on terminal count	When terminal count is reached, an interrupt request generated. This function is extremely useful for generation of real-time clocks.
Programmable one-shot	Output goes low upon receipt of an external trigger edge or software command and returns high when terminal count is reached. This function is retriggerable.
Rate generator	Divide by N counter. The output will go low for one input clock cycle, and the period from one low going pulse to the next is N times the input clock period.
Square-wave rate generator	Output will remain high until 1/2 the count has been completed, and go low for the other half of the count.
Software triggered strobe	Output remains high until software loads count (N). N periods after count is loaded, output goes low for one input clock period.
Hardware triggered strobe	Output goes low for one clock period N counts after rising edge counter trigger input. The counter is retriggerable.
Event counter	On a jumper selectable basis, the clock input becomes an input from the external system. CPU may read the number of events occurring after the counter "window" has been enabled or an interrupt may be generated after N events occur in the system.

Table 4. iSBC® 186/51 Programmable Interrupt Modes

Mode	Operation
Fully nested	Interrupt request line priorities fixed at 0 as highest, 7 as lowest.
Special fully nested	Allows multiple interrupts from slave PICs to the master PIC. Used in the case of cascading where the priority has to be conserved within each slave.
Specific priority	System software assigns lowest priority level. Priority of all other levels based in sequence numerically on this assignment.
Polled	System software examines priority-encoded system interrupt status via interrupt status register.

Table 5. Interrupt Request Sources

Device	Function	Number of Interrupts
MULTIBUS® interface	Requests from MULTIBUS® resident peripherals or other CPU	2
8274	Transmit buffer empty, receive buffer full and channel errors	8
Internal 80186 PIC	Timer 0, 1, 2 outputs (function determined by timer mode) and 2 DMA channel interrupts	5
82586 LCC	Communications processor needs attention	1
Flag byte interrupt	Flag byte interrupt set by MULTIBUS master	1
Systick	80130, MRX system timer	1
Edge to level trigger	Converts EDGE interrupts to level interrupts	1
iSBX™ connectors MULTIMODULE™	Function determined by iSBX™	4 (2 per iSBX connector)
Bus fail safe timer	Indicates addressed MULTIBUS® resident device has not responded to command within 6 msec	1
OR-gate matrix	Outputs of OR-gates on-board for multiple interrupts	1

I/O FUNCTIONALITY

Local Communications Controller

The 82586 is a local communications controller designed to relieve the iAPX 186 of many of the tasks associated with controlling a local network. The 82586 provides most of the functions normally associated with the data link and physical link layers of a local network architecture. In particular, it performs framing (frame boundary delineation, addressing, and bit error detection), link management, and data modulation. It also supports a network management interface.

The iAPX 186 and the 82586 communicate entirely through a shared memory space. To the user, the 82586 appears as two independent but communicating units: the Command Unit (CU) and the Receive Unit (RU). The CU executes the commands given by

the iAPX 186 to the 82586. The RU handles all activities related to packet reception, address recognition, CRC checking, etc. The two are controlled and monitored by the CPU via a shared memory structure called the System Control Block (SCB). Commands for the CU and RU are placed into the SCB by the host processor. Status information is placed into the SCB by the CU and RU (via the CU). The Channel Attention and Interrupt lines are used by the CU and the 82586 to get the other to look into the SCB. See Figure 2.

The 82586 features a high level diagnostic or maintenance capability. It automatically gathers statistics on CRC errors, frame alignment errors, overrun errors, and frames lost due to lack of reception resources. In addition, the user can output the status of all internal registers to facilitate system design.

Upon initialization, the 82586 obtains the address of its System Control Block through the Initialization Root which begins at location 0FFFFF6H. See Figure

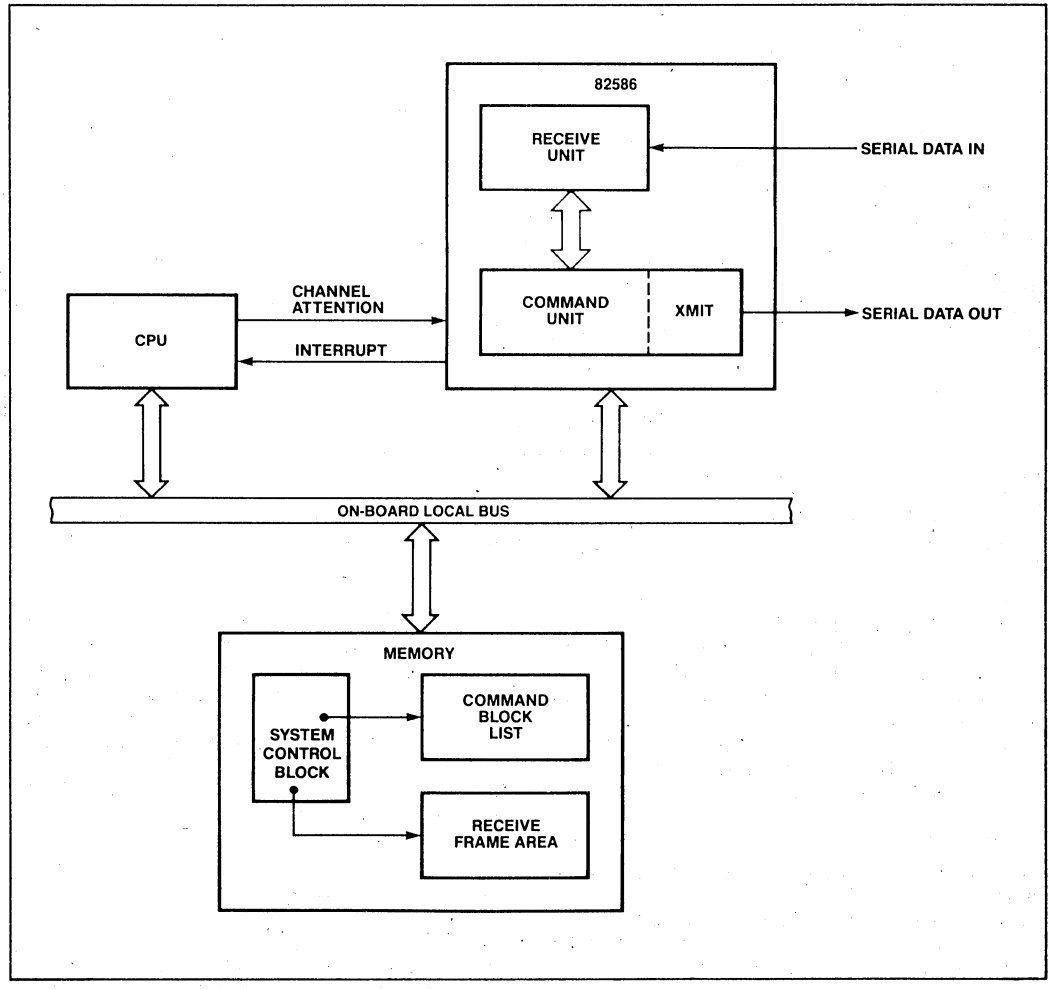


Figure 2. System Overview

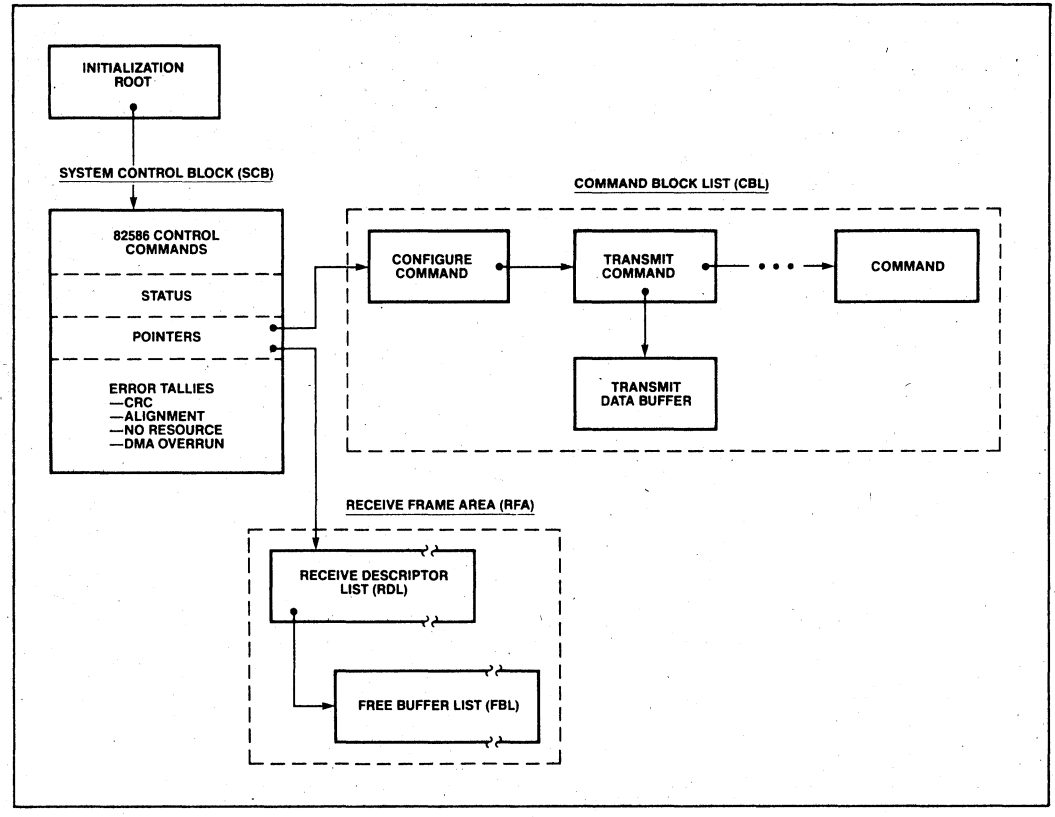


Figure 3. 82586 Memory Structures

3. The SCB contains control commands, status register, pointers to the Command Block List (CBL) and Receive Frame Area (RFA), and tallies for CRC, Alignment, DMA Overrun and No Resource errors. Through the SCB, the 82586 is able to provide status and error counts for the iAPX 86, execute "programs" contained in the CBL and receive incoming frames in the Receive Frame Area (RFA).

Serial I/O

Two programmable communications interfaces using the Intel 8274 Multi-Protocol Serial Controller (MPSC) are contained on the iSBC 186/51. Two independent software selectable BAUD rate generators provide the channels with all the common communications frequencies. The mode of operation (i.e., Asynchronous, Byte Synchronous or Bisynchronous protocols), data format, control character format, parity, and baud rate are all under program control. The 8274 provides full duplex, double buffered transmit and receive capability. Parity, overrun, and framing error detection are all incorporated in the MPSC. The iSBC 186/51 supports operation in the polled, interrupt and DMA driven interfaces through jumper options. The board comes factory defaulted with channel A in RS-422A/RS-449, channel B in RS-232C. Channel A can be configured to support RS-232C also.

iSBX™ MULTIMODULE™ On-Board Expansion

Two 8/16-bit iSBX MULTIMODULE connectors are provided on the iSBC 186/51 microcomputer. Through these connectors, additional on-board I/O functions

may be added. iSBX MULTIMODULE boards optimally support functions provided by VLSI peripheral components such as additional parallel and serial I/O, analog I/O, small mass storage device controllers (e.g., cassettes and floppy disks), and other custom interfaces to meet specific needs. By mounting directly on the single board computer, less interface logic, less power, simpler packaging, higher performance, and lower cost results when compared to other alternatives such as MULTIBUS form factor compatible boards. The iSBX connectors on the iSBC 186/51 boards provide all signals necessary to interface to the local on-board bus, including 16 data lines for maximum data transfer rates. iSBC MULTIMODULE boards designed with 8-bit data paths and using the 8-bit iSBX connector are also supported on the iSBC 186/51 microcomputers. A broad range of iSBX MULTIMODULE options are available in this family from Intel. Custom iSBX modules may also be designed for use on the iSBC 186/51 boards. An iSBX bus interface specification and iSBX connectors are available from Intel.

MEMORY FUNCTIONALITY

RAM Capabilities

The iSBC 186/51 COMMputer board contains 128K Bytes of dual-port dynamic RAM. The on-board RAM may be expanded to 256K Bytes with the iSBC 304 MULTIMODULE board mounted onto the iSBC 186/51 board. The dual-port controller allows access to the on-board RAM (including RAM MULTIMODULE options) from the iSBC 186/51 board and from any other MULTIBUS master via the system bus. Segments of on-board RAM may be configured as a private resource, protected from MULTIBUS system access. The amount of memory allocated as a private resource may be configured in increments of 25% of the total on-board memory ranging from 0% to 100% (optional RAM MULTIMODULE board doubles the increment size). These features allow the multiprocessor systems to establish local memory for each processor and shared system memory configurations where the total system memory size (including local on-board memory) can exceed one megabyte without addressing conflicts.

Universal Memory Sites for Local Memory

Six 28-pin sockets are provided for the use of Intel's 2732, 2764, 27128, 27256 EPROMs and their respective ROMs. When using the 27256s, the on-board EPROM capacity is 192K Bytes. Other JEDEC standard pinout devices are also supported, including byte-wide static RAMs and iRAMs.

MULTIBUS® SYSTEM BUS AND MULTIMASTER CAPABILITIES

Overview

The MULTIBUS system bus is Intel's industry standard microcomputer bus structure. Both 8 and 16-bit single board computers are supported on the MULTIBUS structure with 24 address and 16 data lines. In its simplest application, the MULTIBUS system bus allows expansion of functions already contained on a single board computer (e.g., memory and digital I/O). However, the MULTIBUS structure also allows very powerful distributed processing configurations with multiple processors and intelligent slave I/O, and peripheral boards capable of solving the most demanding microcomputer applications. The MULTIBUS system bus is supported with a broad array of board level products, LSI interface components, detailed published specifications and application notes.

Expansion Capabilities

Memory and I/O capacity may be expanded and additional functions added using Intel MULTIBUS compatible expansion boards. Memory may be expanded by adding user specified combinations of RAM boards, EPROM boards, or combination boards. Input/output capacity may be added with digital I/O and analog I/O expansion boards. Mass storage capability may be achieved by adding single or double density diskette controllers, or hard disk controllers. Modular expandable backplanes and cardcages are available to support multiboard systems.

Multimaster Capabilities

For those applications requiring additional processing capacity and the benefits of multiprocessing (i.e., several CPU's and/or controllers logically sharing system tasks through communication of the system bus), the iSBC 186/51 boards provide full MULTIBUS arbitration control logic. This control logic allows up to *three iSBC 186/51 boards* or other bus masters, including iSBC 80 family MULTIBUS compatible 8-bit single board computers to share the system bus using a serial (daisy chain) priority scheme. This allows up to 16 masters to share the MULTIBUS system bus with an external parallel priority decoder. In addition to the multiprocessing configurations made possible with

multimaster capability, it also provides a very efficient mechanism for all forms of DMA (Direct Memory Access) transfers.

MISCELLANEOUS FUNCTIONALITY

Power-Fail Control and Auxiliary Power

An active-low TTL compatible memory protect signal is brought out on the auxiliary connector which, when asserted, disables read/write access to RAM memory on the board. This input is provided for the protection of RAM contents during system power-down sequences. An auxiliary power bus is also provided to allow separate power to RAM for systems requiring battery back-up of read/write memory. Selection of this auxiliary RAM power bus is made via jumpers on the board.

System Development Capabilities

The development cycle of iSBC 186/51 products can be significantly reduced and simplified by using either the System 86/3XX or the Intellec Series Microcomputer Development Systems. The Assembler, Locating Linker, Library Manager, Text Editor and System Monitor are all supported by the ISIS-II disk-based operating system. To facilitate conversion of 8080A/8085A assembly language programs to run on the iSBC 186/51 boards, CONV-86 is available under the ISIS-II operating system.

In-Circuit Emulator

The Integrated Instrumentation In-Circuit Emulator (I^2ICE) provides the necessary link between the software development environment provided by the Intellec system and the "target" iSBC 186/51 execution system. In addition to providing the mechanism for loading executable code and data into the iSBC 186/51 boards, the I^2ICE-186 provides a sophisticated command set to assist in debugging software and final integration of the user hardware and software.

PL/M-86 and C-86

Intel has two systems implementation languages, PL/M-86 and C-86. Both are standard in the System 86/3XX and are also available as Intellec Microcomputer Development System options. PL/M-86 provides the capability to program in algorithmic language and eliminates the need to manage register usage or allocate memory while still allowing explicit control of the system's resources when needed. C-86 is especially appropriate in applications requiring portability and code density. FORTRAN 86 and PASCAL 86 are also available on Intellec or 86/3XX systems.

Run-Time Support

Intel also offers two run-time support packages: iRMX 88 Realtime Multitasking Executive and the iRMX 86 Operating System. The iRMX 88 executive is a simple, highly configurable and efficient foundation for small, high performance applications. Its multitasking structure establishes a solid foundation for modular system design and provides task scheduling and management, intertask communication and synchronization, and interrupt servicing for a variety of peripheral devices. Other configurable options include terminal handlers, disk file system, debuggers and other utilities. The iRMX 86 Operating System is a highly functional operating system with a very rich set of features and options based on an object-oriented architecture. In addition to being modular and configurable, functions beyond the nucleus include a sophisticated file management and I/O system, and a powerful human interface. Both packages are easily customized and extended by the user to match unique requirements.

SPECIFICATIONS

Word Size

Instruction—8, 16, 24, or 32 bits
Data—8, 16 bits

System Clock

8.00 MHz ± 0.1%

Cycle Time

Basic Instruction Cycle

8 MHz—750 ns
—250 ns (assumes instruction in the queue)

Note: Basic instruction cycle is defined as the fastest instruction time (i.e., two clock cycles.)

Memory Response Time

	Max Access Time	Min Cycle Time
RAM	—	750ns
Universal Memory Sites	200ns	500ns
(jumper selectable)	300ns	625ns

Memory Capacity/Addressing

Six Universal Memory Sites support JEDEC 24/28 pin EPROM, PROM, iRAM and static RAM.

Example for EPROM:

Device	Total Capacity	Address Range
2732	24K Bytes	$F8000\text{-}FFFFF_H$
2764	48K Bytes	$F0000\text{-}FFFFF_H$
27128	96K Bytes	$E0000\text{-}FFFFF_H$
27256	192K Bytes	$C0000\text{-}FFFFF_H$

On-Board RAM

Board	Total Capacity	Address Range
iSBC 186/51	128K Bytes	$0\text{-}1FFFF_H$

With Multimodule™ RAM

Board	Total Capacity	Address Range
iSBC 304	256K Bytes	$0\text{-}3FFFF_H$

I/O Capacity

Serial—two programmable channels using one 8274 iSBX™ Multimodule™—two 8/16-bit iSBX™ connectors allow use of up to 2 single-wide modules or 1 single-wide module and 1 double-wide iSBX module.

Serial Communications Characteristics

Synchronous —5-8 bit characters; internal or external character synchronization; automatic sync insertion

Asynchronous —5-8 bit characters; break character generation; 1, 1/2, or 2 stop bits; false start bit detection

Baud Rates

Frequency (KHz) (S/W Selectable)	Baud Rate (Hz)		
	Synchronous	Asynchronous	
	÷1	÷16	÷64
153.6	—	9600	2400
76.8	—	4800	1200
38.4	38,400	2400	600
19.2	19,200	1200	300
9.6	9,600	600	150
4.8	4,800	300	75
2.4	2,400	150	—
1.76	1,760	110	2400

NOTE:
Frequency selected by I/O write of appropriate 16-bit frequency factor to baud rate register (80186 timer 0 & 80130 baud timer).

Timers

Input Frequencies

Reference: 2 MHz ± 0.1% (.5 μSec period nominal)
Event Rate: 2 MHz max.

80186 Output Frequencies/Timing Intervals

Function	Single Timer/Counter		Dual (Cascaded) Timer/Counter	
	Min	Max	Min	Max
Real-time Interrupt	1.00 μs	65.535ms	6.00 μs	71.580 minutes
Programmable one-shot	1.50 μs	65.535ms	6.00 μs	71.580 minutes
Rate generator	2.342 Hz	1 MHz	.00023 Hz	333.333 KHz
Square-wave rate generator	2.342 Hz	1 MHz	.00023 Hz	333.333 KHz
Software triggered strobe	1.50 μs	65.535ms	6.50 μs	71.580 minutes
Event counter	—	2.00 MHz	—	—

Interfaces

Ethernet—IEEE 802.3 compatible
MULTIBUS®—IEEE 796 compatible
MULTIBUS®—Master D16 M24 I16 V0 EL

Compliance

iSBX™ Bus—IEEE P959 compatible
Serial I/O—RS-232C compatible, configurable as a data set or data terminal, RS-422A/RS-449

Connectors

Interface	Double-Sided Pins	Centers (in.)	Mating Connectors
Ethernet	10	0.1	AMP87531-5
MULTIBUS® SYSTEM	86 (P1)	0.156	Viking 3KH43/9AMK12 Wire Wrap
	60 (P2)	0.1	Viking 3KH30/9JNK
iSBX™ Bus 8-Bit Data	36	0.1	iSBX™ 960-5
16-Bit Data	44	0.1	iSBX™ 960-5
Serial I/O	26	0.1	3M 3452-0001 Flat or AMP88106-1 Flat

Physical Characteristics

Width—12.00 in. (30.48 cm)
Height—6.75 in. (17.15 cm)
Depth—0.70 in. (1.78 cm)
Weight—18.7 ounces

Environmental Characteristics

Operating Temperature—0°C to 55°C
Relative Humidity—10% to 90% (without condensation)

Electrical Characteristics

DC Power Supply Requirements

Configuration	Maximum Current (All Voltages ± 5%)		
	+5	+12	−12
SBC 186/51 as shipped:			
Board Total	6.70A	50mA	45mA
With separate battery back-up	5.90A	50mA	45mA
Battery back-up	.80A	—	—
With SBC-304 Memory Module Installed:			
Board Total	6.80A	50mA	45mA
With separate battery back-up	5.90A	50mA	45mA
Battery back-up	.90A	—	—

NOTES:
1. Add 150 mA to 5V current for each device installed in the 6 available Universal Memory Sites.
2. Add 500 mA to 12V current if Ethernet transceiver is connected.
3. Add additional currents for any SBX modules installed.

Reference Manual

122136-001—iSBC 186/51 Hardware Reference Manual (NOT SUPPLIED)

Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

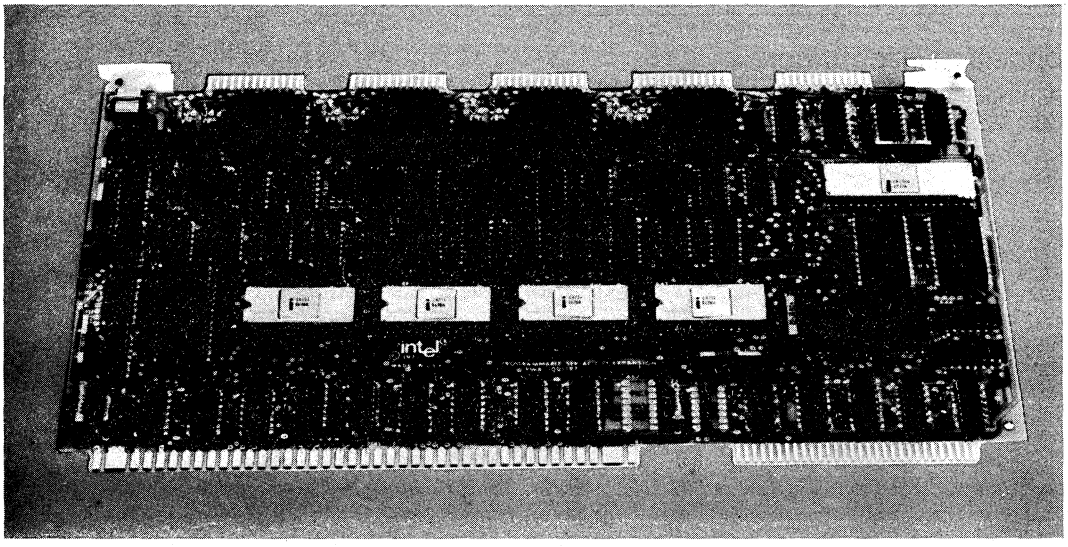
ORDERING INFORMATION

Part Number	Description
SBC 186/51	Communicating Computer

iSBC® 534 (or pSBC 534*)
FOUR CHANNEL COMMUNICATION EXPANSION BOARD

- **Serial I/O expansion through four programmable synchronous and asynchronous communications channels**
- **Individual software programmable baud rate generation for each serial I/O channel**
- **Two independent programmable 16-bit interval timers**
- **Sixteen maskable interrupt request lines with priority encoded and programmable interrupt algorithms**
- **Jumper selectable interface register addresses**
- **16-bit parallel I/O interface compatible with Bell 801 automatic calling unit**
- **RS232C/CCITT V.24 interfaces plus 20 mA optically isolated current loop interfaces (sockets)**
- **Programmable digital loopback for diagnostics**
- **Interface control for auto answer and auto originate modems**

The iSBC 534 Four Channel Communication Expansion Board is a member of Intel's complete line of memory and I/O expansion boards. The iSBC 534 interfaces directly to any single board computer via the MULTIBUS to provide expansion of system serial communications capability. Four fully programmable synchronous and asynchronous serial channels with RS232C buffering and provision for 20 mA optically isolated current loop buffering are provided. Baud rates, data formats, and interrupt priorities for each channel are individually software selectable. In addition to the extensive complement of EIA Standard RS232C signals provided, the iSBC 534 provides 16 lines of RS232C buffered programmable parallel I/O. This interface is configured to be directly compatible with the Bell Model 801 automatic calling unit. These capabilities provide a flexible and easy means for interfacing Intel iSBC based systems to RS232C and optically isolated current loop compatible terminals, cassettes, asynchronous and synchronous modems, and distributed processing networks.



*Same product, manufactured by Intel Puerto Rico, Inc.

FUNCTIONAL DESCRIPTION

Communications Interface

Four programmable communications interfaces using Intel's 8251A Universal Synchronous/Asynchronous Receiver/Transmitter (USART) are contained on the board.* Each USART can be programmed by the system software to individually select the desired asynchronous or synchronous serial data transmission technique (including IBM Bisync). The mode of operation (i.e., synchronous or asynchronous), data format, control character format, parity, and baud rate are all under program control. Each 8251A provides full duplex, double buffered transmit and receive capability. Parity, overrun, and framing error detection are all incorporated in each USART. Each set of RS232C command lines, serial data lines, and signal ground lines are brought out to 26-pin edge connectors that mate with RS232C flat or round cables.

16-Bit Interval Timers

The iSBC 534 provides six fully programmable and independent BCD and binary 16-bit interval timers utilizing two Intel 8253 programmable interval timers.* Four timers are available to the systems designer to generate baud rates for the USARTs under software control. Routing for the outputs from the other two counters is jumper selectable. Each may be independently routed to the programmable interrupt controller to provide real time clocking or to the USARTs (for applications requiring different transmit and receive baud rates). In utilizing the iSBC 534, the systems designer simply configures, via software, each timer independently to meet system requirements. Whenever a given baud rate or time delay is needed, software commands to the programmable timers select the desired function. Three functions of these timers are supported on the iSBC 534, as shown in Table 1. The contents of each counter may be read at any time during system operation.

Table 1. Programmable Timer Functions

Function	Operation
Interrupt on terminal count	When terminal count is reached an interrupt request is generated. This function is used for the generation of real-time clocks.
Rate generator	Divide by N counter. The output will go low for one input clock cycle and high for N – 1 input clock periods.
Square wave rate generator	Output will remain high for one-half the count and low for the other half of the count.

Interrupt Request Lines

Two independent Intel 8259A programmable interrupt controllers (PIC's) provide vectoring for 16 interrupt levels.* As shown in Table 2, a selection of three priority processing algorithms is available to the system designer. The manner in which requests are serviced may thus be configured to match system requirements. Priority assignments may be reconfigured dynamically via software at any time during system operation. Any combination of interrupt levels may be masked through storage, via software, of a single byte to the interrupt mask register of each PIC. Each PIC's interrupt request

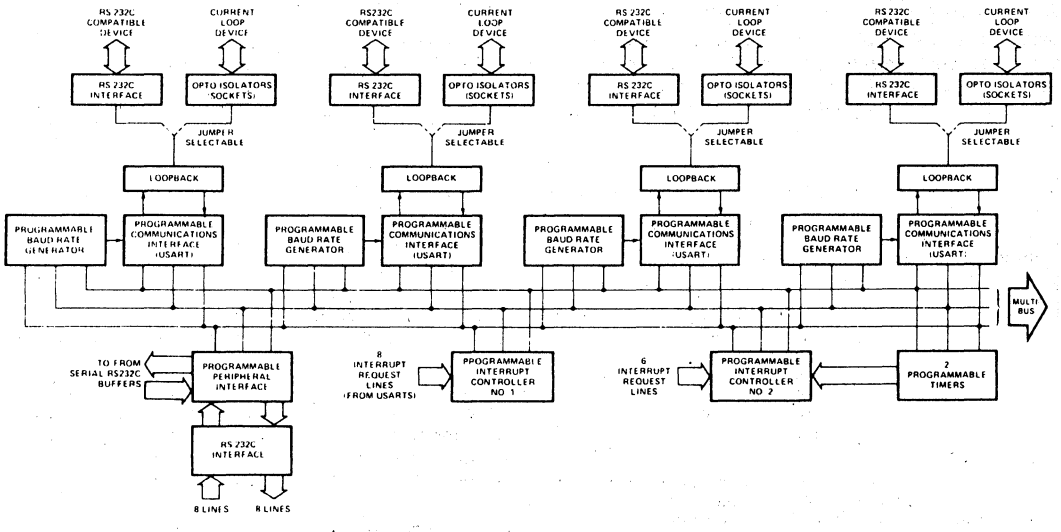


Figure 1. iSBC® 534 Four Channel Communications Expansion Board Block Diagram

output line may be jumper selected to drive any of the nine interrupt lines on the MULTIBUS.

Table 2. Interrupt Priority Options

Algorithm	Operation
Fully nested	Interrupt request line priorities fixed at 0 as highest, 7 as lowest.
Auto-rotating	Equal priority. Each level, after receiving service, becomes the lowest priority level until next interrupt occurs.
Specific priority	System software assigns lowest priority level. Priority of all other levels based in sequence numerically on this assignment.

Table 3. Interrupt Assignments

Interrupt Request Line	PIC 0	PIC 1
0	PORT 0 R_X RDY	PIT 1 counter 1
1	PORT 0 T_X RDY	PIT 2 counter 2
2	PORT 1 R_X RDY	Ring indicator (all ports)
3	PORT 1 T_X RDY	Present next digit
4	PORT 2 R_X RDY	Carrier detect port 0
5	PORT 2 T_X RDY	Carrier detect port 1
6	PORT 3 R_X RDY	Carrier detect port 2
7	PORT 3 T_X RDY	Carrier detect port 3

Interrupt Request Generation — As shown in Table 3, interrupt requests may originate from 16 sources. Two jumper selectable interrupt requests (8 total) can be automatically generated by each USART when a character is ready to be transferred to the MULTIBUS system bus (i.e., receive buffer is full) or a character has been transmitted (transmit buffer is empty). Jumper selectable requests can be generated by two of the programmable timers (PITs), and six lines are routed directly from peripherals to accept carrier detect (4 lines), ring indicator, and the Bell 801 present next digit request lines.

Systems Compatibility

The iSBC 534 provides 16 RS232C buffered parallel I/O lines implemented utilizing an Intel 8255A programmable peripheral interface (PPI) configured to operate in mode 0.* These lines are configured to be directly compatible with the Bell 801 automatic calling unit (ACU). This capability allows the iSBC 534 to interface to Bell 801 type ACUs and up to four modems or other serial communications devices. For systems not requiring interface to an ACU, the parallel I/O lines may also be used as general purpose RS232C compatible control lines in system implementation.

* Complete operational details on the Intel 8251A USART, the Intel 8253 Programmable Interval Timer, the Intel 8255A Programmable Peripheral Interface, and the Intel 8259A Programmable Interrupt Controller are contained in the Intel Component Data Catalog.

SPECIFICATIONS

Serial Communications Characteristics

Synchronous — 5-8 bit characters; internal or external character synchronization; automatic sync insertion.
Asynchronous — 5-8 bit characters; break character generation; 1, 1½, or 2 stop bits; false start bit detection.

Sample Baud Rates[1]

Frequency[2] (kHz, Software Selectable)	Baud Rate (Hz)		
	Synchronous	Asynchronous	
		÷ 16	÷ 64
153.6	—	9600	2400
76.8	—	4800	1200
38.4	38400	2400	600
19.2	19200	1200	300
9.6	9600	600	150
4.8	4800	300	75
6.98	6980	—	110

Notes:
1. Baud rates shown here are only a sample subset of possible software-programmable rates available. Any frequency from 18.75 Hz to 614.4 kHz may be generated utilizing on-board crystal oscillator and 16-bit programmable interval timer (used here as frequency divider).
2. Frequency selected by I/O writes of appropriate 16-bit frequency factor to Baud Rate Register.

Interval Timer and Baud Rate Generator Frequencies

Input Frequency (On-Board Crystal Oscillator) — 1.2288 MHz ± 0.1% (0.813 μs period, nominal)

Function	Single Timer		Dual/Timer Counter (Two Timers Cascaded)	
	Min	Max	Min	Max
Real-Time Interrupt Interval	1.63 μs	53.3 ms	3.26 μs	58.25 minutes
Rate Generator (Frequency)	18.75 Hz	614.4 kHz	0.0029 Hz	307.2 kHz

Interfaces — RS232C Interfaces
EIA Standard RS232C Signals provided and supported:

Carrier detect	Receive data
Clear to send	Ring indicator
Data set ready	Secondary receive data
Data terminal ready	Secondary transmit data
Request to send	Transmit clock
Receive clock	Transmit data

Parallel I/O — 8 input lines, 8 output lines, all signals RS232C compatible
Bus — All signals MULTIBUS system bus compatible

I/O Addressing

The USART, interval timer, interrupt controller, and parallel interface registers of the iSBC 534 are configured as a block of 16 I/O address locations. The location of this block is jumper selectable to begin at any 16-byte I/O address boundary (i.e., 00H, 10H, 20H, etc.).

I/O Access Time

400 ns USART registers
400 ns Parallel I/O registers
400 ns Interval timer registers
400 ns Interrupt controller registers

Compatible Connectors

Interface	Pins (qty.)	Centers (in.)	Mating Connectors
Bus	86	0.156	Viking 2KH43/9 AMK12
Serial and parallel I/O	26	0.1	3M 3462-0001 or TI H312113

Compatible Opto-Isolators

Function	Supplier	Part Number
Driver	Fairchild General Electric Monsanto	4N33
Receiver	Fairchild General Electric Monsanto	4N37

Physical Characteristics

Width — 12.00 in. (30.48 cm)
Height — 6.75 in. (17.15 cm)
Depth — 0.50 in. (1.27 cm)
Weight — 14 oz (398 gm)

Electrical Characteristics

Average DC Current

Voltage	Without Opto-Isolators	With Opto-Isolators[1]
V_{CC} = +5V	1.9 A, max	1.9 A, max
V_{DD} = +12V	275 mA, max	420 mA, max
V_{AA} = −12V	250 mA, max	400 mA, max

Note

1. With four 4N33 and four 4N37 opto-isolator packages installed in sockets provided to implement four 20 mA current loop interfaces.

Environmental Characteristics

Operating Temperature — 0 °C to +55 °C

Reference Manual

502140-002 — iSBC 534 Hardware Reference Manual (NOT SUPPLIED)

Reference manuals are shipped with each product only if designated SUPPLIED (see above). Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

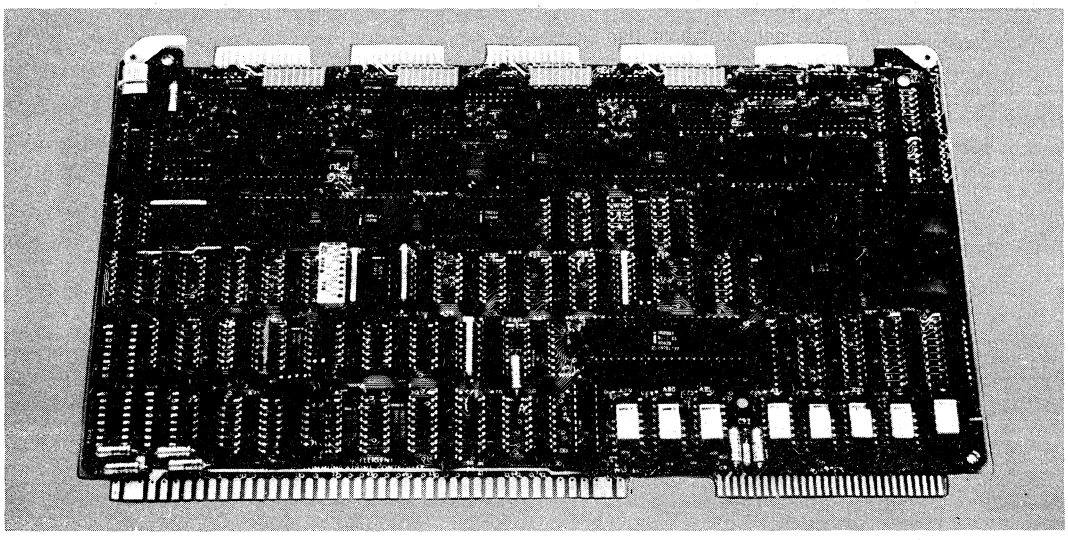
ORDERING INFORMATION

Part Number	Description
SBC 534	Four Channel Communication Expansion Board

iSBC® 544
INTELLIGENT COMMUNICATIONS CONTROLLER

- **iSBC® Communications Controller acting as a single board communications computer or an intelligent slave for communications expansion**
- **On-board dedicated 8085A Microprocessor providing communications control and buffer management for four programmable synchronous/asynchronous channels**
- **Sockets for up to 8K bytes of EPROM**
- **16K bytes of dual port dynamic read/write memory with on-board refresh**
- **Extended MULTIBUS® addressing permits iSBC 544 board partitioning into 16K-byte segments in a 1-megabyte address space**
- **Ten programmable parallel I/O lines compatible with Bell 801 Automatic Calling Unit**
- **Twelve levels of programmable interrupt control**
- **Individual software programmable baud rate generation for each serial I/O channel**
- **Three independent programmable interval timer/counters**
- **Interface control for auto answer and auto originate modem**

The iSBC 544 Intelligent Communications Controller is a member of Intel's family of single-board computers, memory, I/O, and peripheral controller boards. The iSBC 544 board is a complete communications controller on a single 6.75 x 12.00 inch printed circuit card. The on-board 8085A CPU may perform local communications processing by directly interfacing with on-board read/write memory, nonvolatile read only memory, four synchronous/asynchronous serial I/O ports, RS232/RS366 compatible parallel I/O, programmable timers, and programmable interrupts.



FUNCTIONAL DESCRIPTION

Intelligent Communications Controller

Two Mode Operation — The iSBC 544 board is capable of operating in one of two modes: 1) as a single board communications computer with all computer and communications interface hardware on a single board; 2) as an "intelligent bus slave" that can perform communications related tasks as a peripheral processor to one or more bus masters. The iSBC 544 may be configured to operate as a stand-alone single board communications computer with all MPU, memory and I/O elements on a single board. In this mode of operation, the iSBC 544 may also interface with expansion memory and I/O boards (but no additional bus masters). The iSBC 544 performs as an intelligent slave to the bus master by performing all communications related tasks. Complete synchronous and asynchronous I/O and data management are controlled by the on-board 8085A CPU to coordinate up to four serial channels. Using the iSBC 544 as an intelligent slave, multichannel serial transfers can be managed entirely on-board, freeing the bus master to perform other system functions.

Architecture — The iSBC 544 board is functionally partitioned into three major sections: I/O, central computer, and shared dual port RAM memory (Figure 1). The I/O hardware is centered around the four Intel 8251A USART devices providing fully programmable serial interfacing. Included here as well is a 10-bit parallel interface compatible with the Bell 801 automatic calling unit, or equivalent. The I/O is under full control of the on-board CPU and is protected from access by system bus masters. The second major segment of the intelligent communications controller is a central computer, with an 8085A CPU providing powerful processing capability. The 8085A together with on-board EPROM / ROM, static RAM, programmable timers/counters, and program-

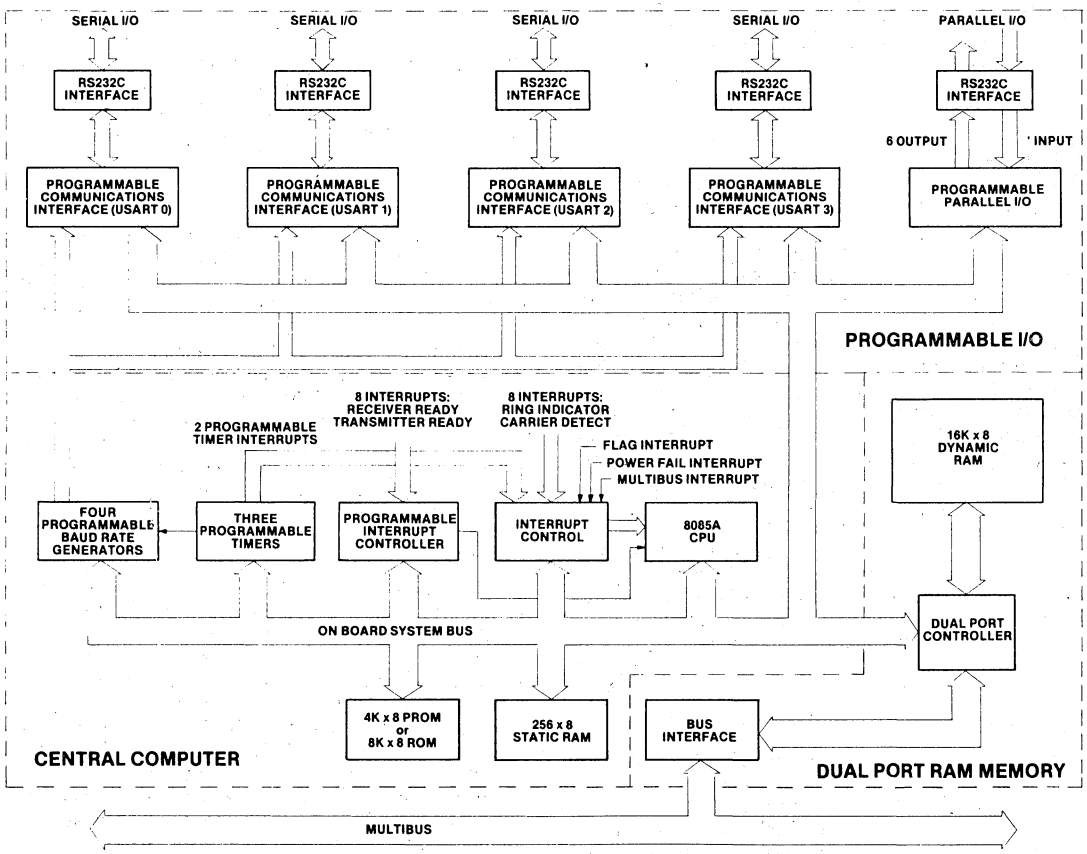


Figure 1. iSBC ® 544 Intelligent Communications Controller Block Diagram

mable interrupt control provide the intelligence to manage sophisticated communications operations on-board the iSBC 544 board. The timer/counters and interrupt control are also common to the I/O area providing programmable baud rates to the USARTs and prioritizing interrupts generated from the USARTs. The central computer functions are protected for access only by the on-board 8085A. Likewise, the on-board 8085A may not gain access to the system bus when being used as an intelligent slave. When the iSBC 544 is used as a bus master, the on-board 8085A CPU controls complete system operation accessing on-board functions as well as memory and I/O expansion. The third major segment, dual port RAM memory, is the key link between the iSBC 544 intelligent slave and bus masters managing the system functions. The dual port concept allows a common block of dynamic memory to be accessed by the on-board 8085A CPU and off-board bus masters. The system program can, therefore, utilize the shared dual port RAM to pass command and status information between the bus masters and on-board CPU. In addition, the dual port concept permits blocks of data transmitted or received to accumulate in the on-board shared RAM, minimizing the need for a dedicated memory board.

Serial I/O

Four programmable communications interfaces using Intel's 8251A Universal Synchronous/Asynchronous Receiver/Transmitter (USART) are contained on the board and controlled by the on-board CPU in combination with the on-board interval timer/counter to provide all common communication frequencies. Each USART can be programmed by the system software to individually select the desired asynchronous or synchronous serial data transmission technique (including IBM Bisync). The mode of operation (i.e., synchronous or asynchronous), data format, control character format, parity, and baud rate are all under program control. Each 8251A provides full duplex, double-buffered, transmit and receive capability. Parity, overrun, and framing error detection are all incorporated in each USART. Each channel is fully buffered to provide a direct interface to RS232C compatible terminals, peripherals, or synchronous/asynchronous modems. Each channel of RS232C command lines, serial data lines, and signal ground lines are brought out to 26-pin edge connectors that mate with RS232C flat or round cable.

Parallel I/O Port

The iSBC 544 provides a 10-bit parallel I/O interface controlled by an Intel 8155 Programmable Interface (PPI) chip. The parallel I/O port is directly compatible with an Automatic Calling Unit (ACU) such as the Bell Model 801, or equivalent, and can also be used for auxiliary functions. All signals are RS232C compatible, and the interface cable signal assignments meet RS366 specifications. For systems not requiring an ACU interface, the parallel I/O port can be used for any general purpose interface requiring RS232C compatibility.

Central Processing Unit

Intel's powerful 8-bit n-channel 8085A CPU, fabricated on a single LSI chip, is the central processor for the iSBC 544. The 8085A CPU is directly software compatible with the Intel 8080A CPU. The 8085A contains six 8-bit general purpose registers and an accumulator. The six general purpose registers may be addressed individually or in pairs, providing both single and double precision operators. The minimum instruction execution time is 1.45 microseconds. The 8085A CPU has a 16-bit program counter. An external stack, located within any portion of iSBC 544 read/write memory, may be used as a last-in/first-out storage area for the contents of the program counter, flags, accumulator, and all of the six general purpose registers. A 16-bit stack pointer controls the addressing of this external stack. This stack provides subroutine nesting bounded only by memory size.

EPROM/ROM Capacity

Sockets for up to 8K bytes of nonvolatile read only memory are provided on the iSBC 544 board. Read only memory may be added in 2K-byte increments up to a maximum of 4K bytes using Intel 2716 EPROMs or masked ROMs; or in 4K-byte increments up to 8K bytes maximum using Intel 2732 EPROMs. All on-board EPROM/ROM operations are performed at maximum processor speed.

RAM Capacity

The iSBC 544 contains 16K bytes of dynamic read/write memory using Intel 2117 RAMs. Power for the on-board RAM may be provided on an auxiliary power bus, and memory protect logic is included for RAM battery back-up requirements. The iSBC 544 contains a dual port controller, which provides dual port capability for the on-board RAM memory. RAM accesses may occur from either the on-board 8085A CPU or from another bus master, when used as an intelligent slave. Since on-board RAM accesses do not require the MULTIBUS, the bus is available for concurrent bus master use. Dynamic RAM refresh is accomplished automatically by the iSBC 544 for accesses originating from either the CPU or from the MULTIBUS.

Addressing — On board RAM, as seen by the on-board 8085A CPU, resides at address 8000_H-$BFFF_H$. On-board RAM, as seen by an off-board CPU, may be placed on any 4K-byte address boundary. The iSBC 544 provides extended addressing jumpers to allow the on-board RAM to reside within a one megabyte address space when accessed via the MULTIBUS. In additon, jumper options are provided which allow the user to protect 8K- or 12K-bytes on-board RAM for use by the on-board 8085 CPU only. This reserved RAM space is not accessible via the MULTIBUS and does not occupy any system address space.

Static RAM — The iSBC 544 board also has 256 bytes of static RAM located on the Intel 8155 PPI. This memory is only accessible to the on-board 8085A CPU and is located at address $7F00_H$-$7FFF_H$.

Programmable Timers

The iSBC 544 board provides seven fully programmable and independent interval timer/counters utilizing two Intel 8253 Programmable Interval Timers (PIT), and the Intel 8155. The two Intel 8253 PITs provide six independent BCD or binary 16-bit interval timer/counters and the 8155 provides one 14-bit binary timer/counter. Four of the PIT timers (BDGO-3) are dedicated to the USARTs providing fully independent programmable baud rates.

Three General Use Timers — The fifth timer (BDG4) may be used as an auxiliary baud rate to any of the four USARTs or may alternatively be cascaded with timer six to provide extended interrupt intervals. The sixth PIT timer/counter (TINT1) can be used to generate interrupt intervals to the on-board 8085A. In addition to the timer/counters on the 8253 PITs, the iSBC 544 has a 14-bit timer available on the 8155 PPI providing a third general use timer/counter (TINT0). This timer output is jumper selectable to the interrupt structure of the on-board 8085A CPU to provide additional timer/counter capability.

Timer Functions — In utilizing the iSBC 544 board, the systems designer simply configures, via software, each timer independently to meet systems requirements. Whenever a given baud rate or interrupt interval is needed, software commands to the programmable timers select the desired function. The on-board PITs together with the 8155 provide a total of seven timer/counters and six operating modes. Mode 3 of the 8253 is the primary operating mode of the four dedicated USART baud rate generators. The timer/counters and useful modes of operation for the general use timer/counters are shown in Table 1.

Table 1. Programmable Timer Functions

Function	Operation	Counter
Interrupt on Terminal Count (Mode 0)	When terminal count is reached, an interrupt request is generated. This function is useful for generation of real-time clocks.	8253 TINT1
Rate Generator (Mode 2)	Divide by N counter. The output will go low for one input clock cycle and high for N-1 input clock periods.	8253 BDG4 *
Square-Wave Rate Generator (Mode 3)	Output will remain high until one-half the TC has been completed, and go low for the other half of the count. This is the primary operating mode used for generating a Baud rate clocked to the USARTs.	8253 BDG0-4 TINT1
Software Triggered Strobe (Mode 4)	When the TC is loaded, the counter will begin. On TC the output will go low for one input clock period.	8253 BDG4 * TINT1
Single Pulse	Single pulse when TC reached.	8155 TINT0
Repetitive Single Pulse	Repetitive single pulse each time TC is reached until a new command is loaded.	8155 TINT0

* BDG4 is jumper selectable as an auxiliary baud rate generator to the USARTs or as a cascaded output to TINT1. BDG4 may be used in modes 2 and 4 only when configured as a cascaded output.

Interrupt Capability

The iSBC 544 board provides interrupt service for up to 21 interrupt sources. Any of the 21 sources may interrupt the intelligent controller, and all are brought through the interrupt logic to 12 interrupt levels. Four interrupt levels are handled directly by the interrupt processing capability of the 8085A CPU and eight levels are serviced from an Intel 8259A Programmable Interrupt Controller (PIC) routing an interrupt request output to the INTR input of the 8085A (see Table 2).

Interrupt Sources — The 22 interrupt sources originate from both on-board communications functions and the Multibus. Two interrupts are routed from each of the four USARTs (8 interrupts total) to indicate that the transmitter and receiver are ready to move a data byte to or from the on-board CPU. The PIC is dedicated to accepting these 8 interrupts to optimize USART service request. One of eight interrupt request lines are jumper selectable for direct interface from a bus master via the system bus. Two auxiliary timers (TINT0 from 8155 and TINT1 from 8253) are jumper selectable to provide general purpose counter/timer interrupts. A jumper selectable Flag Interrupt is generated to allow any bus master to interrupt the iSBC 544 by writing into the base address of the shared dual port memory accessable to the system. The Flag Interrupt is then cleared by the iSBC 544 when the on-board processor reads the base address. This interrupt provides an interrupt link between

Table 2. Interrupt Vector Memory Locations

Interrupt Source		Vector Location	Interrupt Level
Power Fail	TRAP	24_H	1
8253 TINT1 8155 TINT0	RST 7.5	$3C_H$	2
Ring Indicator (1) Carrier Detect	RST 6.5	34_H	3
Flag Interrupt INT0/-INT7/ (1 of 8)	RST 5.5	$2C_H$	4
RXRDY0 TXRDY0 RXRDY1 TXRDY1 RXRDY 2 TXRDY2 RXRDY3 TXRDY3	INTR	Programmable	5-12

(1) Four ring indicator interrupts and four carrier detect interrupts are summed to the RST 6.5 input. The 8155 may be interrogated to inspect any one of the eight signals.

a bus master and intelligent slave (See System Programming). Eight inputs from the serial ports are monitored to detect a ring indicator and carrier detect from each of the four channels. These eight interrupt sources are summed to a single interrupt level of the 8085A CPU. If one of these eight interrupts occur, the 8155 PPI can then be interrogated to determine which port caused the interrupt. Finally, a jumper selectable Power Fail Interrupt is available from the Multibus to detect a power down condition.

8085 Interrupt — Thirteen of the twenty-two interrupt sources are available directly to four interrupt inputs of the on-board 8085A CPU. Requests routed to the 8085A interrupt inputs, TRAP, RST 7.5, RST 6.5 and RST 5.5 have a unique vector memory address. An 8085A jump instruction at each of these addresses then provides software linkage to interrupt service routines located independently anywhere in the Memory. All interrupt inputs with the exception of the TRAP may be masked via software.

8259A Interrupts — Eight interrupt sources signaling transmitter and receiver ready from the four USARTs are channeled directly to the Intel 8259A PIC. The PIC then provides vectoring for the next eight interrupt levels. Operating mode and priority assignments may be reconfigured dynamically via software at any time during system operation. The PIC accepts transmitter and receiver interrupts from the four USARTs. It then determines which of the incoming requests is of highest priority, determines whether this request is of higher priority than the level currently being serviced, and, if appropriate, issues an interrupt to the CPU. The output of the PIC is applied directly to the INTR input of the 8085A. Any combination of interrupt levels may be masked, via software, by storing a single byte in the interrupt mask register of the PIC. When the 8085A responds to a PIC interrupt, the PIC will generate a CALL instruction for each interrupt level. These addressses are equally spaced at intervals of 4 or 8 (software selectable) bytes. Interrupt response to the PIC is software programmable to a 32- or 64-byte block of memory. Interrupt sequences may be expanded from this block with a single 8085A jump instruction at each of these addresses.

Interrupt Output — In addition, the iSBC 544 board may be jumper selected to generate an interrupt from the on-board serial output data (SOD) of the 8085A. The SOD signal may be jumpered to any one of the 8 MULTIBUS interrupt lines (INT0/-INT7/) to provide an interrupt signal directly to a bus master.

Power-Fail Control

Control logic is also included to accept a power-fail interrupt in conjunction with the AC-low signal from the iSBC 635 Power Supply or equivalent.

Expansion Capabilities

When the iSBC 544 board is used as a single board communications controller, memory and I/O capacity may be expanded and additional functions added using Intel MULTIBUS™ compatible expansion boards. In this mode, no other bus masters may be configured in the system. Memory may be expanded to a 65K byte capacity by adding user specified combinations of RAM boards, EPROM boards, or combination boards. Input/output capacity may be increased by adding digital I/O and analog I/O expansion boards. Furthermore, multiple iSBC 544 boards may be included in an expanded system using one iSBC 544 board as a single board communications computer and additional controllers as intelligent slaves.

System Programming

In the system programming environment, the iSBC 544 board appears as an additional RAM memory module when used as an intelligent slave. The master CPU communicates with the iSBC 544 board as if it were just an extension of system memory. Because the iSBC 544 board is treated as memory by the system, the user is able to program into it a command structure which will allow the iSBC 544 board to control its own I/O and memory operation. To enhance the programming of the iSBC 544 board, the user has been given some specific tools. The tools are: 1) the flag interrupt, 2) an on-board RAM memory area that is accessible to both an off-board CPU and the on-board 8085A through which a communications path can exist, and 3) access to the bus interrupt line.

Flag Interrupt — The Flag Interrupt is generated anytime a write command is performed by an off-board CPU to the base address of the iSBC 544 board's RAM. This interrupt provides a means for the master CPU to notify the iSBC 544 board that it wishes to establish a communications sequence. In systems with more than one intelligent slave, the flag interrupt provides a unique interrupt to each slave outside the normal eight MULTIBUS interrupt lines (INT0/-INT7/).

On-Board RAM — The on-board 16K byte RAM area that is accessible to both an off-board CPU and the on-board 8085A can be located on any 4K boundary in the system. The selected base address of the iSBC 544 RAM will cause a flag interrupt when written into by an off-board CPU.

Bus Access — The third tool to improve system operation as an intelligent slave is access to the Multibus interrupt lines. The iSBC 544 board can both respond to interrupt signals from an off-board CPU, and generate an interrupt to the off-board CPU via the MULTIBUS.

System Development Capability

The development cycle of iSBC 544 board based products may be significantly reduced using the Intellec series microcomputer development systems. The Intellec resident macroassembler, text editor, and system monitor greatly simplify the design, development and debug of iSBC 544 system software. An optional ISIS-II diskette operating system provides a linker, object code locater, and library manager. A unique in-circuit emulator (ICE-85) option provides the capability of developing and debugging software directly on the iSBC 544 board.

SPECIFICATIONS

Serial Communications Characteristics

Synchronous — 5-8 bit characters; automatic sync insertion; parity.

Asynchronous — 5-8 bit characters; break character generation; 1, 1½, or 2 stop bits; false start bit detection; break character detection.

Baud Rates

Frequency (KHz)[1] (Software Selectable)	Baud Rate (Hz)[2]		
	Synchronous	Asynchronous	
		÷ 16	÷ 64
153.6	— —	9600	2400
76.8	— —	4800	1200
38.4	38400	2400	600
19.2	19200	1200	300
9.6	9600	600	150
4.8	4800	300	75
6.98	6980	— —	110

Notes:
1) Frequency selected by I/O writes of appropriate 16-bit frequency factor to Baud Rate Register.
2) Baud rates shown here are only a sample subset of possible software programmable rates available. Any frequency from 18.75 Hz to 614.4 KHz may be generated utilizing on-board crystal oscillator and 16-bit Programmable Interval Timer (used here as a frequency divider).

8085A CPU

Word Size — 8, 16 or 24 bits/instruction; 8 bits of data

Cycle Time — 1.45/usec ± .1% for fastest executable instruction; i.e. four clock cycles.

Clock Rate — 2.76 MHz ± .1%

System Access Time

Dual port memory — 740 nsec

Note: Assumes no refresh contention

Memory Capacity

On-Board ROM/PROM — 4K, or 8K bytes of user installed ROM or EPROM.

On-Board Static RAM — 256 bytes on 8155.

On-Board Dynamic RAM (on-board access) — 16K bytes. Integrity maintained during power failure with user-furnished batteries (optional).

On-Board Dyanmic RAM (MULTIBUS access) — 4K, 8K, or 16K-bytes available to bus by switch selection.

Memory Addressing

On-Board ROM/PROM — 0-0FFF (using 2716 EPROMs or masked ROMs); 0-1FFF (using 2732A EPROMs)

On-Board Static Ram — 256 bytes: 7F00-7FFF

On-Board Dynamic RAM (on-board access) — 16K bytes: 8000-BFFF.

On-Board Dynamic RAM (MULTIBUS access) — any 4K increment 00000-FF000 which is switch and jumper selectable. 4K- 8K- or 16K-bytes can be made available to the bus by switch selection.

I/O Capacity

Serial — 4 programmable channels using four 8251A USARTs.

Parallel — 10 programmable lines available for Bell 801 ACU, or equivalent use. Two auxiliary jumper selectable signals.

I/O Addressing

On-Board Programmable I/O

Port	Data	Control
USART 0	D0	D1
USART 1	D2	D3
USART 2	D4	D5
USART 3	D6	D7
8155 PPI	E9 (Port A) EA (Port B) EB (Port C)	E8

Interrupts

Addresses for 8259A Registers (Hex notation, I/O address space)

E6 Interrupt request register
E6 In-service register
E7 Mask register
E6 Command register
E7 Block address register
E6 Status (polling register)

Note: Several registers have the same physical address: Sequence of access and one data bit of the control word determines which register will respond.

Interrupt levels routed to the 8085 CPU automatically vector the processor to unique memory locations:

24 TRAP
3C RST 7.5
34 RST 6.5
2C RST 5.5

Timers

Addresses for 8253 Registers (Hex notation, I/O address space)

Programmable Interrupt Timer One

D8	Timer 0	BDG0
D9	Timer 1	BDG1
DA	Timer 2	BDG2
DB	Control register	

Programmable Interrupt Timer Two

DC	Timer 0	BDG3
DD	Timer 1	BDG4
DE	Timer 2	TINT1
DF	Control register	

Address for 8155 Programmable Timer

E8 Control
ED Timer (LSB) TINT0
EC Timer (MSB) TINT0

Input frequencies — Jumper selectable reference 1.2288 MHz ± .1% (.814 usec period nominal) or 1.843 MHz ± .1% crystal (0.542 usec period, nominal)

Output Frequencies (at 1.2288 MHz)

Function	Single timer/counter		Dual timer/counter (two timers cascaded)	
	Min	Max	Min	Max
Real-time interrupt interval	1.63 usec	53.3 usec	3.26 usec	58.25 min
Rate Generator (frequency)	18.75 Hz	614.4 KHz	0.00029 Hz	307.2 KHz

Interfaces

Serial I/O — EIA Standard RS232C signals provided and supported:

Carrier Detect
Clear to Send
Data Set Ready
Data Terminal Ready
Request to Send
Receive Clock
Receive Data
Ring Indicator
Secondary Receive Data *
Secondary Transmit Data *
Transmit Clock
Transmit Data
DTE Transmit Clock

* Optional if parallel I/O port is not used as Automatic Calling Unit.

Parallel I/O — Four inputs and eight outputs (includes two jumper selectable auxiliary outputs). All signals compatible with EIA Standard RS232C. Directly compatible with Bell Model 801 Automatic Calling Unit, or equivalent.

MULTIBUS — Compatible with iSBC MULTIBUS.

On-Board Addressing

All communications to the parallel and serial I/O ports, to the timers, and to the interrupt controller, are via read and write commands from the on-board 8085A CPU.

Auxiliary Power

An auxiliary power bus is provided to allow separate power to RAM for systems requiring battery backup of read/write memory. Selection of this auxiliary RAM power bus is made via jumpers on the board.

Connectors

Interface	Pins (qty)	Centers (in.)	Mating Connectors
Bus	86	0.156	Viking 2KH43/9AMK12
Parallel I/O	50	0.1	3M 3415-000 or AMP 88083-1
Serial I/O	26	0.1	3M 3462-000 or AMP 88373-5

Memory Protect

An active-low TTL compatible memory protect signal is brought out on the auxiliary connector which, when asserted, disables read/write access to RAM memory on the board. This input is provided for the protection of RAM contents during the system power-down sequences.

Bus Drivers

Function	Characteristic	Sink Current (mA)
Data	Tri-state	50
Address	Tri-state	15
Commands	Tri-state	32

Note: Used as a master in the single board communications computer mode.

Physical Characteristics

Width: 30.48 cm (12.00 inches)
Depth: 17.15 cm (6.75 inches)
Thickness: 1.27 cm (0.50 inch)
Weight: 3.97 gm (14 ounces)

Electrical Characteristics

DC Power Requirements

	Current Requirements			
Configuration	V_{CC} = +5V ±5% (max)	V_{DD} = ±12V ±5% (max)	V_{BB} = −5V(3) ±5% (max)	V_{AA} = −12V ±5% (max)
With 4K EPROM (using 2716)	I_{CC} = 3.4 max max	I_{DD} = 350mA max	I_{BB} = 5mA max	I_{AA} = 200mA max
Without EPROM	3.3A max	350 mA max	5 mA max	200 mA max
RAM only (1)	390 mA max	176 mA max	5 mA max	—
RAM(2) refresh only	390 mA max	20 mA max	5 mA max	

Notes: 1. For operational RAM only, for AUX power supply rating.
2. For RAM refresh only. Used for battery backup requirements. No RAM accessed.
3. V_{BB} is normally derived on-board from V_{AA}, eliminating the need for a V_{BB} supply. If it is desired to supply V_{BB} from the bus, the current requirement is as shown.

Environmental Characteristics

Operating Temperature: 0°C to 55°C (32°F to 131°F)
Relative Humidity: To 90% without condensation

Reference Manual

502160 — iSBC 544 Intelligent Communications Controller Board Hardware Reference Manual (NOT SUPPLIED)

Reference manuals are shipped with each product only if designated SUPPLIED (see above). Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

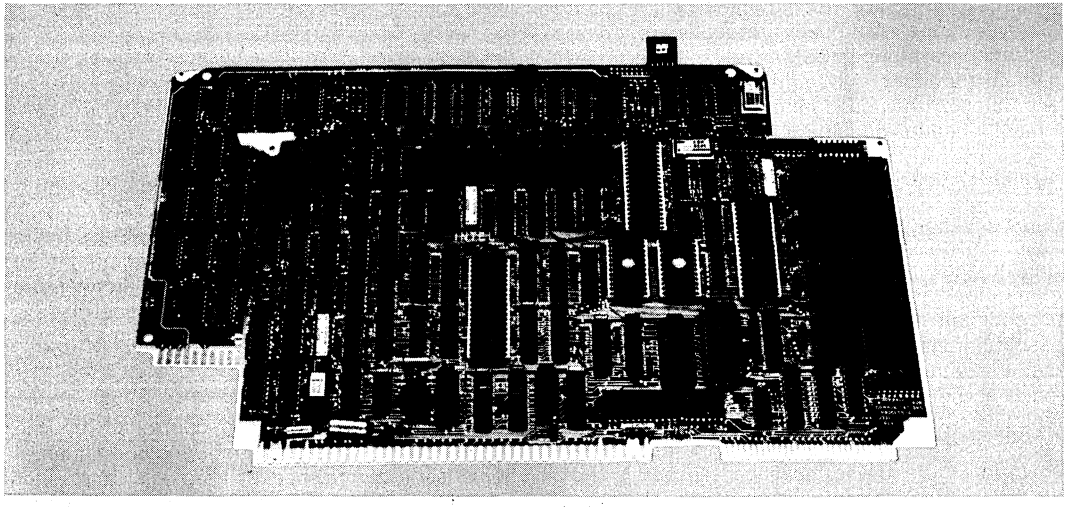
ORDERING INFORMATION

Part Number	Description
iSBC 544	Intelligent Communications Controller

ISBC® 550
ETHERNET COMMUNICATIONS CONTROLLER

- **Meets the version 1.0 tri-corporate Ethernet specification**
- **Ethernet data link layer support**
  - **Data encapsulation**
  - **Framing and packet control**
  - **Buffer management**
- **Ethernet physical link layer support**
  - **Serial/deserialization**
  - **10 Mbits per second data rate**
  - **CRC generation/check**
  - **Carrier-sense multiple-access with collision detection (CSMA/CD)**
  - **Transceiver interface compatibility**
- **Easy-to-use MULTIBUS® interprocessor protocol supported in firmware**
- **Power-up confidence test assures integrity of on-board memory and programmable LSI**
- **Traffic, errors and collision information maintained for network management**
- **Excellent foundation for Ethernet local area end-to-end network**

The iSBC 550 Ethernet Communications Controller meets the tri-corporate (DEC, Xerox, Intel) specification for Ethernet local area networks. All the functions of the Ethernet data link layer and physical link layer are provided on two 6.75 × 12″ circuits boards and associated firmware. The MULTIBUS compatible controller can be utilized as the foundation for a single board computer (iSBC)-based Ethernet local area network or as a prototype for Intel® 8085, iAPX™ 88, or iAPX 86 component-based Ethernet applications. The iSBC 550 controller's firmware (supplying the Ethernet and system interface) has an easy-to-use MULTIBUS Interprocessor Protocol (MIP) facility, which is readily accessed from another iSBC Board using a custom run-time software system or Intel's iRMX™ 80/88/86 Real-Time Executive software and the iMMX™ 800 (MULTIBUS Message Exchange) software package. The Ethernet data link functions are divided between the processor board which provides the data link layer's software to control the data encapsulation and the link management, and the serial/deserialization (SerDes) board which provides the 10-MBit per second serial interface to the Ethernet transceiver.



FUNCTIONAL DESCRIPTION

The iSBC 550 Ethernet Communications Controller is a two-board MULTIBUS-compatible set that offers high-speed Ethernet-compatible data transfer between digital devices operating at a 10-Mbit per sec data rate. The iSBC 550 controller can effectively support the needs of local area network applications, such as office automation, distributed data processing, factory data collection, research data collection, intelligent terminal and other EDP-related products.

Ethernet Specification

The Ethernet network is a local area network concept that is jointly being supported by Intel Corporation, Digital Equipment Corporation, and Xerox Corporation. The network is designed to link systems over a distance of up to 2500 meters using an available 50-ohm coaxial cable. Several hundred stations may be connected to the cable which supports a data rate of 10 Megabits per second. The data is encapsulated in a packet message format. The data signal is a base-band, Manchester-encoded type that is self-synchronizing.

The jointly developed Ethernet specification, "The Ethernet, A Local Area Network Data Link Layer and Physical Link Layer Specification, Version 1.0, September 30, 1980", precisely defines the two lower layers of a local area network architecture where the system is a series of independent layers. The lowest layer, the physical link layer, is concerned with coaxial cable interface. The data link layer supports the peer protocol's statistical contention resolution (CSMA/CD) and link management functions. All additional network layers are defined by the user during the implementation of the application-specific layers.

Ethernet Data Link Layer Support

The iSBC 550 processor board provides the data link layer's software to control the data encapsulation and the link management, including frame delimitation, address handling, error detection, and collision handling. After the iSBC 550 processor board is initialized upon system start-up or reset, the data link firmware is ready to service the local area network commands. An example of a command structure sent the iSBC controller to receive a packet of data from the Ethernet link is shown in Figure 1. The message passed via the MIP (MULTIBUS Interprocessor Protocol) interface is composed of two parts, the iSBC 550 controller information (including the command and associated data), and the required Ethernet information.

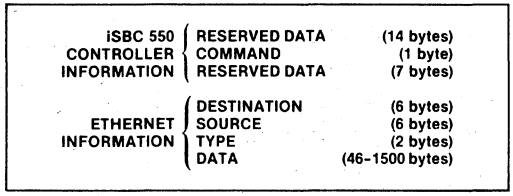


Figure 1. Data Link for SUPPLYBUF Command Format

Shown in Table 1 are eight external Ethernet controller commands available to a user's application via the MIP interface. The commands manage the Ethernet multicast address recognition, message type connection, message flow, and overall network statistics.

Table 1. External Controller Commands

Command	Function
CONNECT	Indicates the data link message TYPE to be *connected* to user program.
DISCONNECT	*Disconnects* the data link TYPE from the user's application.
ADDMCID	*Adds* a multicast ID for recognition.
DELETEMCID	*Delete* the specified multicast ID.
TRANSMIT	*Transmit* a data packet to the Ethernet link.
SUPPLYBUF	*Supplies* a buffer for packet reception from the Ethernet link ("receive" function).
READ	*Read* the statistical variables maintained by data link layer.
READC	*Read* and *clear* the statistical variables.

Ethernet Physical Link Layer Support

The Serialization/Deserialization (SerDes) board provides the required electrical characteristics of the physical link layer of the Ethernet architecture for a transceiver interface. The transceiver is a device physically attached to the coax cable which does signal conditioning for transmitting and receiving.

Many major functions are controlled by the SerDes board. These functions include serialization/deserialization, packet framing, Manchester encoding/decoding, transmit data flow control, receive data flow control, destination address decoding for received message, CRC generation and

checking, and diagnostics for CRC error, loopback, transmit timeout, and CSMA/CD (Carrier-Sense Multiple-Access with Collision-Detection).

Easy-To-Use Interface

One of the iSBC 550 controller boards is an iAPX 88-based processor board which has firmware support for the user's application interface. The programmatic interface utilizes the MULTIBUS Interprocessor Protocol (MIP) interface to the processor board. This interface is concerned with the message-passing protocol between multiple-processors. The iMMX 800 (MULTIBUS Message Exchange) software supports the MIP interface and offers a convenient quick-start method for users of Intel's iRMX 80, iRMX 88 executives and iRMX 86 operating system products for an Ethernet-based application.

Confidence Test

An effective diagnostic function is implemented in firmware on the processor board. This function is invoked at system initialization during both power-up and system reset time. These functions include: packet CRC checking, memory test, controller loopback, and other error tests. The tests provide a fundamental level of controller integrity.

Network Statistics

Statistics maintained by the data link firmware include packet traffic counts, collision information and error totals. This information can be effectively utilized by the user's application to understand the network's operation.

End-To-End Networking Foundation

The iSBC 550 controller provides the foundation data link layer and the physical link layer for a local area network architecture. Typically, the higher levels are user-defined and include the transport and the session control layers. The transport control layer is concerned with the end-to-end communications and the virtual channel connection via a port-to-port address. The session control layer provides the process-to-process control function which includes symbolic name binding and the establishment of the virtual connection via the transport control layer. In addition, the session control provides the specific error and recovery control responsible for message delivery.

The higher levels of the local area network architecture (see Figure 2) which use the data link layer are outside of the Ethernet standard, but can be implemented quickly on companion iSBC boards (e.g., iSBC 80/24, iSBC 88/25, iSBC 86/12A) running under the iRMX 80/88/86 Real-Time Multitasking Executives, respectively, and associated iMMX MULTIBUS Message Exchange (iMMX 800) software. Special iSBC 550 device driver software compatible with the iRMX 86 and iRMX 88 file systems is provided in the iMMX 800 package.

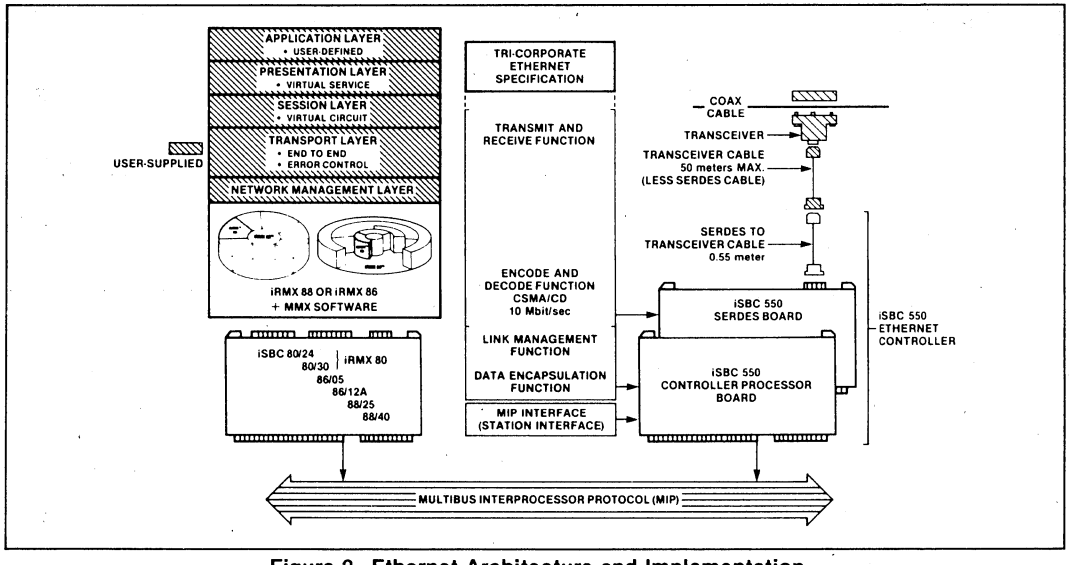


Figure 2. Ethernet Architecture and Implementation

SPECIFICATIONS

Memory Addressing Capability

MULTIBUS System Bus — (00000-EFFFF)

Ethernet I/O Channels

One Ethernet electrically-compatible transceiver line on the SerDes board.

Interface Specifications

MULTIBUS System Bus — All signals TTL compatible.

Transceiver — All signals Ethernet specifications transceiver compatible.

Serial Communications Characteristics

Bit Serial Frame — Provides 64-bit *preamble*, 48-bit *destination* address, 48-bit *source* address, 16-bit *type*, 46-1500 bytes for *data*, and a *frame* check sequence of 32 bits.

Ethernet Network Specifications Supported

Coax Cable Length — 500-meter max.
Transceiver Cable Length — 50-meter max.
Number of Stations — 100 max.
Baud Rate — 10-Mbit/sec

System Clock

5.00 MHz, ±0.1%

Physical Characteristics (Both Boards)

Width — 12.00 in. (30.48 cm) (each board)
Height — 6.75 in. (17.15 cm) (each board)
Depth — 0.5 in. (1.27 cm) (each board)
Weight — 3.5 lb (1.6 kg) (both boards)

SerDes to Transceiver Cable

Length — 0.55 meter (22 in.). Four pair twisted-wire cable with SerDes connector and transceiver interface connector.

Electrical Characteristics

Power requirements for both boards
+5 VDC @ 9.0A max..
+12 VDC @ 0.5A max.

Environmental Characteristics

Operating Temperature — 0°C to 55°C

Relative Humidity — To 90% (without condensation)

Connectors

Interface	Pins (qty)	Centers (in.)	Mating Connectors
MULTIBUS System	86	0.156	Viking 2KH43/9AMK12
SerDes Edge Connector	10	0.1	AMP 87631-5 Housing AMP 87195-9 Pins
Transceiver	15	0.1	Cinch Type DA 51220-1

Reference Manuals

121746 — iSBC 550 Ethernet Communications Controller Hardware Reference Manual (NOT SUPPLIED)

121769 — The Ethernet Communications Controller Programmer's Reference Manual (NOT SUPPLIED)

Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, CA 95051.

ORDERING INFORMATION

Part Number	Description
SBC 550	Ethernet Communications Controller for 10 Mbit/sec coaxial transmission. Includes Ethernet data link control software and cable to transceiver.

PRELIMINARY

iSBC® 570, 576, 577
INTEL SPEECH TRANSACTION FAMILY

- **Friendly man-machine interface—speech is the most natural and most easily learned form of interaction for man.**
- **Lower data entry cost—source data capture**
- **Higher accuracy—operator mental encoding is eliminated.**
- **Freedom of Movement—More efficient work flow**
- **Hands and eyes free—ability to perform another primary task**
- **Easier training—interactive, generic terminology**
- **Complements keyboard/CRT—new dimension to data entry**

Users world wide are recognizing the many advantages of having Automatic Speech Recognition (ASR) and Electronic Speech Synthesis (ESS) in their products and applications. Speech I/O is a new dimension in data entry/control that complements other I/O mechanisms.

Speech I/O as a direct man-machine interface can be used for a broad range of applications, such as office and factory automation, computer-aided design, QC inspection stations, inventory control—and many more. Whatever your application is, the benefits of speech I/O are measured in dollars saved, improved productivity and improved product quality.

FEBRUARY 1983
ORDER NUMBER: 210598

In computer-aided design and manufacturing (CAD/CAM), design commands by speech allow the design engineer to keep his attention focused on the actual graphic elements.

In manufacturing, speech transactions provide important advantages in productivity. Defect tracing, production line monitoring and synchronization, and factory data collection, all benefit from direct human speech to computer communication.

In the automated office, ever-increasing machine intelligence can be controlled without mastering of typing skills.

The basic concept of a speech I/O system is shown in Figure 1. The speech I/O system provides a human-oriented interface with a machine-oriented computer-based information system or process. The speech I/O system recognizes speech inputs, provides visual/audio prompts and verification, and handles message editing and buffering. Depending on what was recognized, digitally coded data is then used to interact with the machine-oriented computer-based system.

The functional blocks of a speech I/O system are shown in Figure 2.

A complete system includes not just the capabilities for signal conditioning, Automatic Speech Recognition (ASR), and Electronic Speech Synthesis (ESS), but must include speech transaction processing as well. The Speech Transaction Processing task includes:

—The conversion between spoken language and coded representation
—Operator prompting and feedback
—Message editing
—Message buffering

In addition, development tools should be available for the generation of speech transaction files that will define the operations of the speech I/O system. Figure 3 shows the function of each member of the Intel Speech Transaction Family.

The Intel Speech Transaction Family, iSBC® 570, iSBC® 576 and iSBC® 577, is a family of products that provides a minimal risk path to add speech Input/Output (I/O) to your product line. The Speech Transaction Family will allow you to move from evaluation to integral speech driven products without major redesigns. Depending on your stage of product development, whether it is an evaluation, or a product simulation, or an add-on speech option, or a

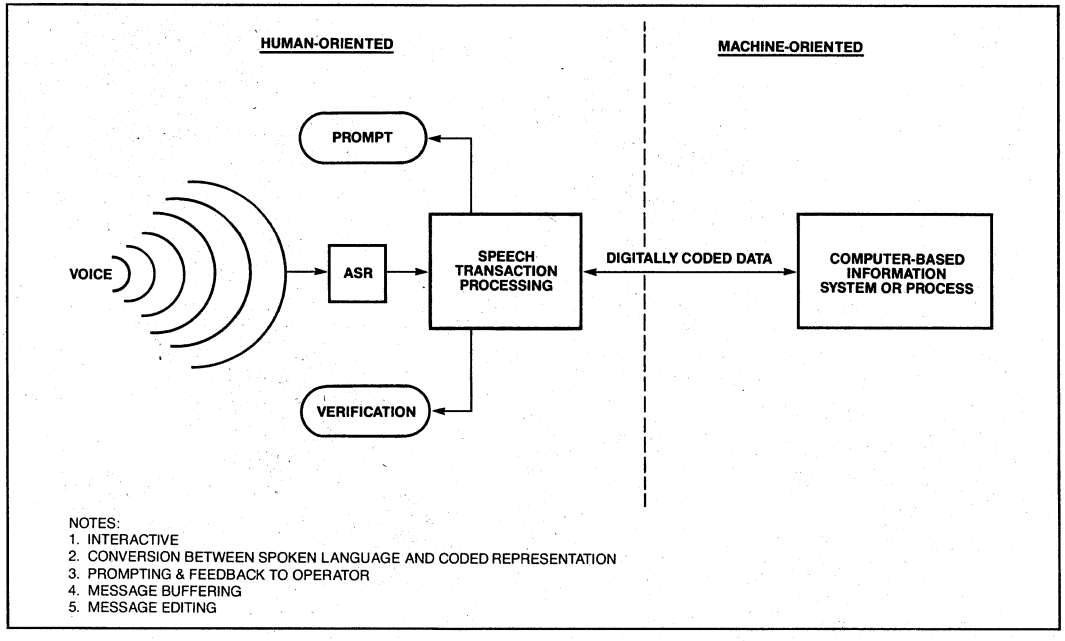


Figure 1. Basic Concept

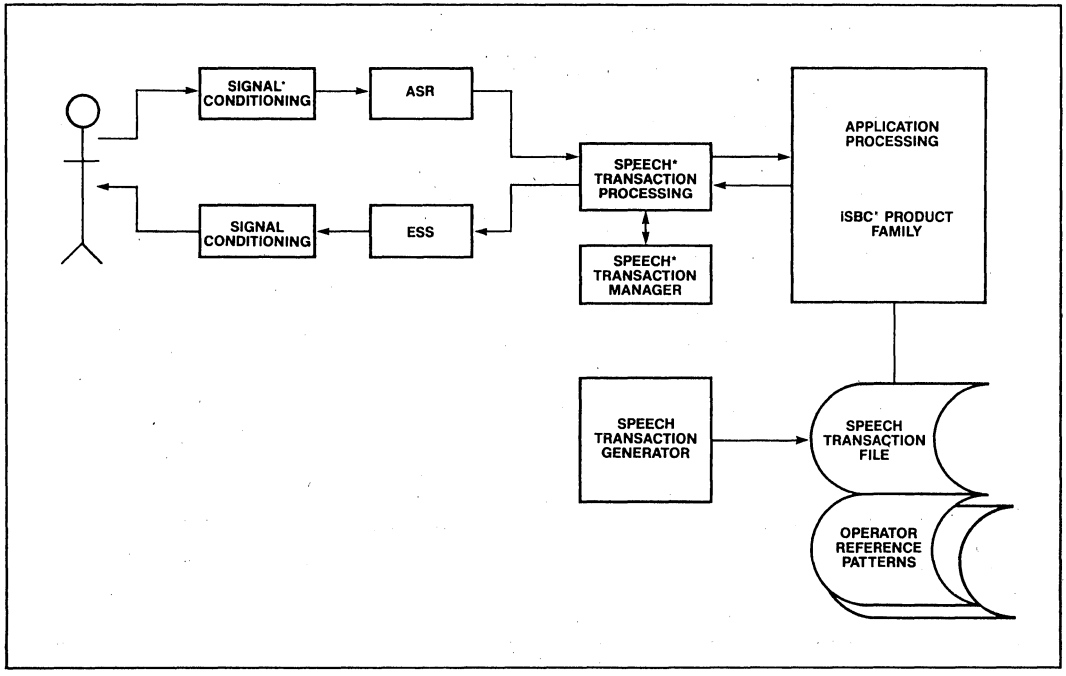


Figure 2. Functional Blocks of Speech I/O System.

fully integrated speech product, the Speech Transaction Family's flexibility allows your speech I/O application to grow with a minimal amount of engineering effort. The Speech Transaction Family allows you to adapt your product to various markets as your application needs change, without a major redesign. Whether it is a configured speech development system, or easy-to-integrate speech board, or a maximum value-added speech component chip set, an Intel product is ready to meet your needs.

Development of your speech I/O system may have been your stumbling block in the past. The requirement for speech technology expertise, extensive hardware development and extensive software development are a thing of the past. Integral to the Speech Transaction Family are highly sophisticated computer-based design and development tools that will take you from product concept to a working speech product with a minimal effort. In-depth knowledge of speech algorithms and of speech human factors considerations are no longer an absolute requirement of your system designers.

Intel provides the total solution. Speech hardware has been designed to work with our wide selection of Multibus® single-board computers, memory cards, and data I/O cards. Speech software is based on the Real-Time Multi-Tasking Executive (RMX-88). Speech transaction software development has been implemented on our universal Intellec® Microcomputer Development System. All of the pieces have been engineered to provide an easily integrated speech I/O solution.

Speech I/O is a new technology area. Intel has developed a family of products and services, that will fit your development sequence needs for a new technology with minimal risk and ease of use. A very likely evaluation and development sequence you may follow is illustrated in Figure 4 and Figure 5 along with Intel's products and services that are offered to meet those needs. Having products and services that can satisfy the illustrated sequence is very important in reducing the risk, engineering cost, and lowering incremental investments necessary as product requirements change.

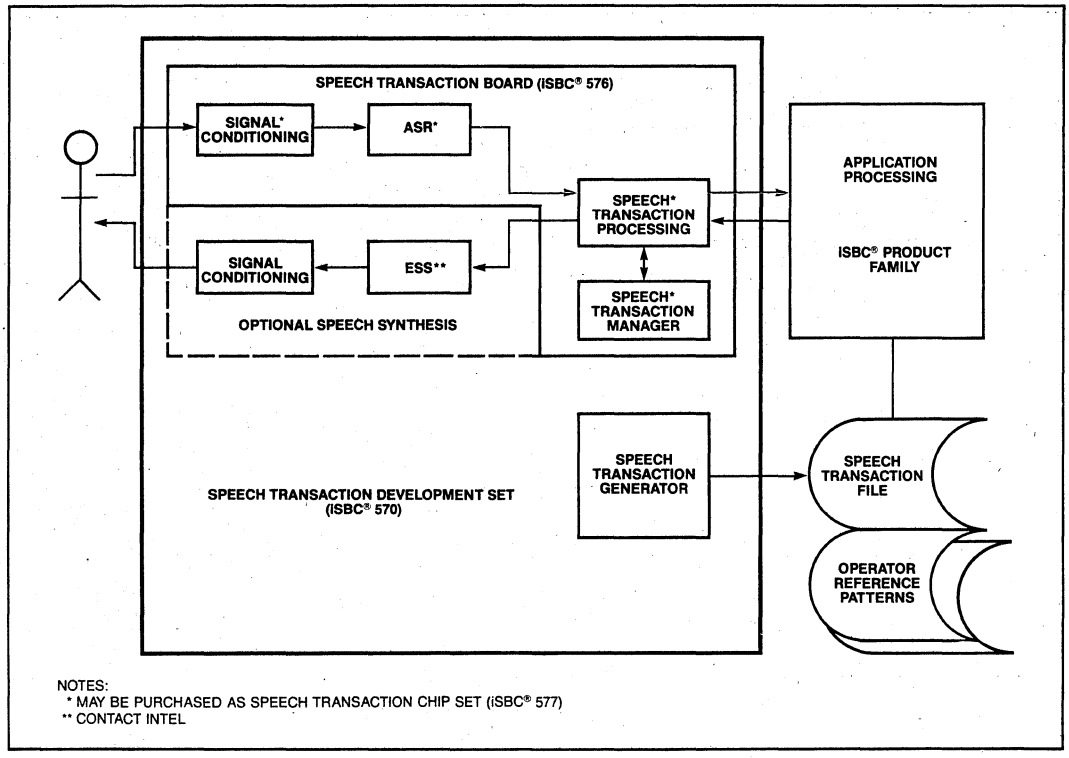


Figure 3. Functional Blocks of the Intel Speech Transaction Family.

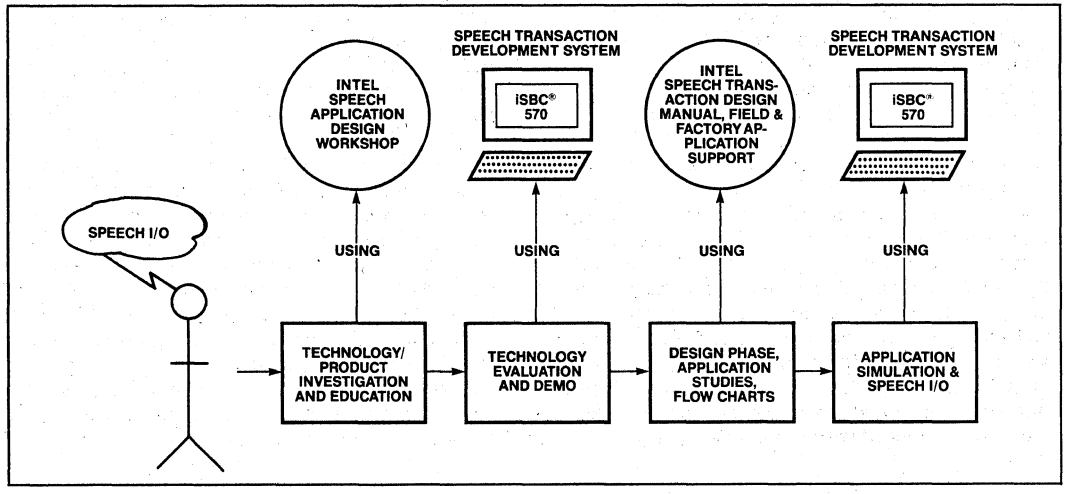


Figure 4. Application Definition Phases

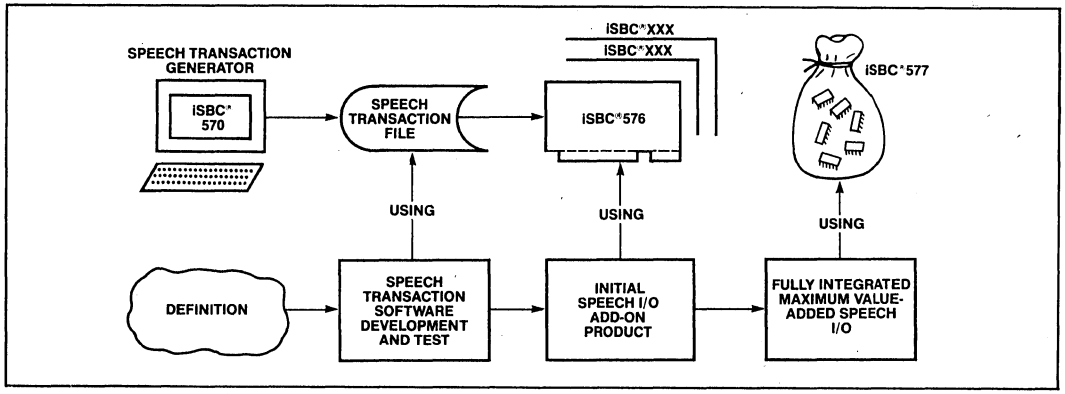


Figure 5. Application Implementation Phases

The sequence starts with a workshop to learn about the Speech Technology and to develop a necessary knowledge base to evaluate potential applications. The next stage, an evaluation-oriented Speech Transaction Development System (iSBC® 570 and Intellec® Microcomputer Development System), provides technology evaluation and demonstrations without engineering investment. Using the experience from the two previous stages, plus field and factory application support, the design phase can now proceed. Once the application framework has been established, application simulation can be performed using the Speech Transaction Development System.

Upon successful completion of simulation, the speech transaction software development can be easily completed on the same Speech Transaction Development System. The initial speech I/O products can then be shipped using the Speech Transaction Board (iSBC® 576). When higher volume justifies increasing the value added, the chip set, iSBC® 577, can be used. Throughout the process, whether it is system, board, or chip set, the same software is utilized. Very little is lost as your product needs change. The level of investment required tracks the stage of product development. Your risk and exposure is kept to a minimum.

SPEECH APPLICATION DESIGN WORKSHOP "SPEECH COMMUNICATION WITH COMPUTERS"

- **4-day "hands-on" experience**
- **Understanding of application design involving speech input**
- **How to evaluate speech recognition performance**
- **Use of Intel speech development tools**
- **Application selection criteria for speech I/O**
- **Interfacing to host system**

The Intel Speech Design Workshop is an intensive four days that will result in an understanding, based on experience, of speech as an input method. The experienced system designer via laboratory and lecture will rapidly develop an understanding of the capabilities and limitations of speech recognition as an input peripheral system. This is an ideal first step toward mastering the exciting new technology of speech transaction processing. All persons directly involved in the selection or implementation of a speech recognition–based system should attend. Attendees are presumed to be familiar with computers, a programming language, and the Intellec® Microcomputer Development System. The student will experience all phases of solution development by designing, implementing, testing, and presenting to the class an extension to the basic speech demonstration provided with the Speech Transaction Development Set.

DAY 1

What's unique about speech?—Communicating with computers using words and phrases rather than keystrokes.

Overview of the Intel Speech Transaction Family.

Introduction to the Speech Transaction Board (STB), Speech Transaction Manager (STM), Speech Transaction File (STF), and the Speech Transaction Generator (STG).

Lab: Install iSBC® 570 Speech Transaction Development Set. Operate Speech Training Demo Program.

The 7 modes of operation of the STM.

Lab: Using the Evaluation mode to ascertain performance.

DAY 2

The Speech Transaction Generator (STG).

The Speech Transaction File (STF) using the training demonstration program as an example.

Application selection criteria.

Transaction design.

Lab: Develop an extension of the training demonstration program using the STG.

DAY 3

Human factors considerations—case histories.

Operator training and performance evaluation.

Automatic Speech Recognition (ASR) subsystem parameters.

Lab: Evaluate performance of new STF developed on Day 2. Measure effects of ASR parameter changes on recognition.

DAY 4

Configuring the application solution.

Host I/O interfacing.

Host program design concepts.

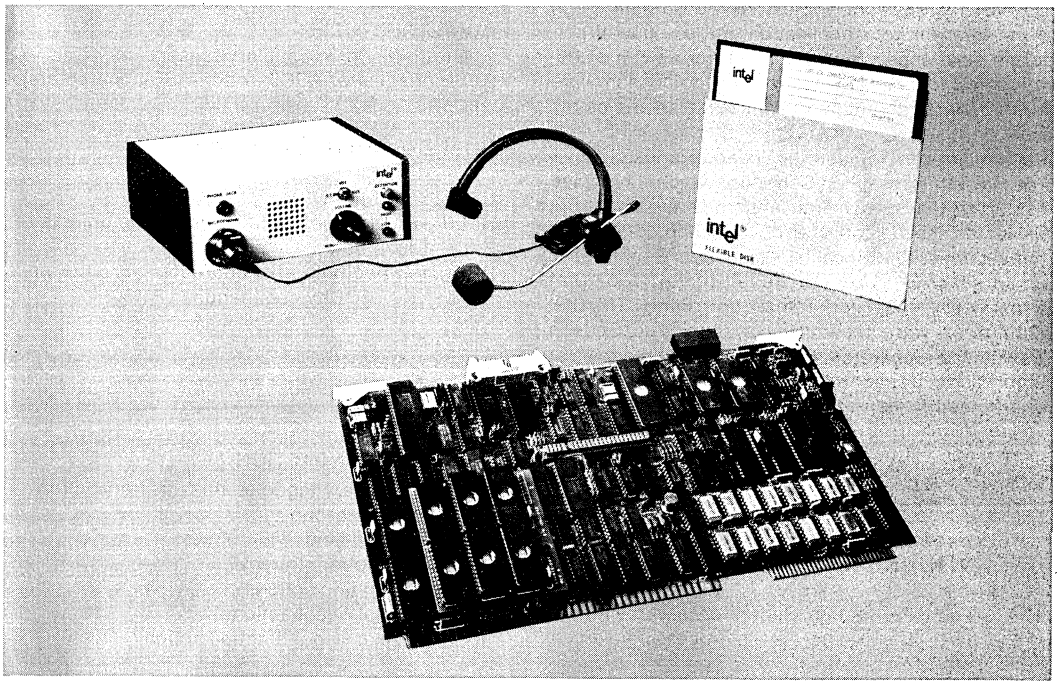
Lab: Discuss student-developed program extensions.

PRELIMINARY

iSBC® 570
SPEECH TRANSACTION DEVELOPMENT SET

- **Complete Development Support Set for the Intel Speech Product Family. Includes:**
  - **—Speech Transaction Generator**
  - **—iSBC® 576 Speech Transaction Board**
  - **—iSBC® 575 Operator Control Unit**
  - **—Microphone**
  - **—Demo program**
  - **—Speech Transaction Design Manual**
- **Intellec® Microcomputer Development System based**
- **Speech Transaction Generator provides:**
  - **—Interactive design environment**
  - **—A speech transaction structure embodying good human factors engineering**
  - **—Automatic error checking of transaction design**
  - **—Symbolic labeling for easy system designer reference**
  - **—Speech Transaction File data base manager facilitates Speech Transaction File changes**

The Speech Transaction Development Set, iSBC® 570, provides an easy-to-use package for speech transaction evaluation, design simulation and application development. Along with Intel's Speech Design Workshop, the Speech Transaction Development Set becomes the starter kit that will move you into the forefront of speech I/O systems. Using the demo program supplied, you are quickly introduced to the important attributes of speech. Using the Intellec® Microcomputer Development System and writing/modifying software based on examples provided, you can quickly simulate your application without hardware development. And finally, with the Speech Transaction Generator, your speech transaction structure, definition, transaction file coding and mangement become a well-defined automated task.



FUNCTIONAL DESCRIPTION

The iSBC® 570 Speech Transaction Development Set has been designed to meet your speech I/O needs as your level of involvement with speech I/O system grows. The Speech Transaction Development Set serves three very important functions. The three functions are: 1) Technology Evaluation and Demonstration, 2) Application Simulation of Speech I/O, and 3) Design and Development of Speech Transaction Software. These three functions are discussed below.

Technology Evaluation and Demonstration — A complete demo package is provided for you to demonstrate the capabilities of speech I/O. This package allows you to evaluate the speech technology without investing engineering design and development time. It is easy to use. Major attributes of a speech I/O system are highlighted and fully documented. The host system for the demonstration is the Intellec® Microcomputer Development System.

Application Simulation of Speech I/O—The Speech Transaction Development Set provides the necessary tools and program examples for you to easily simulate your speech I/O system using the Intellec® Microcomputer Development System as the host. With the iSBC® 570 and the Intellec® Microcomputer Development System, you can now design a speech I/O system for your application and see how it performs. Your speech transaction structure can be developed and checked out without doing hardware and software integration with the rest of your system.

Design and Development of Speech Transactions — The Speech Transaction Generator which is provided as part of the Speech Transaction Development Set facilitates the design and development of speech transactions. The Speech Transaction Generator is an interactive software development tool that generates the Speech Transaction File (STF) that configures your speech I/O system. The Speech Transaction Generator checks for inconsistencies or incomplete transactions. The generated code is guaranteed to be fully compatible with the Speech Transaction Board. The Speech Transaction Generator will not only shorten your development time, but will also facilitate a well human-engineered speech I/O interface.

OPERATIONAL DESCRIPTION

The Speech Transaction Generator is implemented in two parts. The first part is the processing element of the STG and resides on EPROM in an STB environment. The second part is the data base manager for the STG and resides as an executable file under ISIS. The STG allows a system designer (with appropriate knowledge of transaction, fields, vocabulary and synthesis) to specify a STF easily. The STG maintains a set of files on the Intellec® system as the data base. In this manner, the STG is the customization tool used by the speech system designers to prepare application-unique speech transactions that will execute on the STB under the supervision of the Speech Transaction Manager (STM). The STG also allows the system designer to dump portions of this data base in an ASCII-text format to a file. This ASCII-text file is useful for transporting data base entries between the STG implemented on other than an ISIS environment.

The things that a system designer can manipulate with the STG are termed "objects." Objects can be catagorized into structures and non-structures. Structures are generally a string of characters or a list of tags. Objects are classified as follows:

STRUCTURES
1. Transaction
2. Fields
3. Vocabulary
4. Synthesis

NON-STRUCTURES
1. Group (list of vocabulary tags)
2. Strings (list of ASCII or non-ASCII characters)

Brief Description of Commands

UTILITY COMMANDS

HELP—Provides information about the objects
EXIT—Close data base and exit STG
PREfix—Specify prefix character for DEFine or MODify commands

EDIT COMMANDS

DEFine

DEFINE TRANSACTION:
1. Vocabulary tag to enable this transaction?
2. Training group?
3. Starting field?
4. Host buffer strategy?
5. Verification actions?
6. Special reject actions?
7. Special illegal function action?

DEFINE FIELD:
1. Prompt?
2. Help message?
3. Prefix for host message?
4. Suffix for host message?
5. Special functions enable?
6. Valid sources?
7. Multiple utterance path?
   If yes,
   a) Vocabulary words?
   b) Next field?
   c) Maximum number of utterances?
   d) Fixed or variable?
8. Vocabulary words?
9. Next field?

DEFINE VOCABULARY:
1. Name?
2. Visual verify?
3. Audio verify?
4. Host message?
5. Visual train?
6. Audio train?
7. Special functions?

DEFINE SYNTHESIS:
1. Function?
2. Duration?
3. Delay?

DELete
Removes objects from the data base.

MODify
Modifies objects already entered into the data base with the DEFine command.

VALIDATION AND GENERATION COMMANDS

VALidate
Sequences through each of the transactions specified and validates them for completeness and proper definition.

GENerate
Takes the result of a successful validate command and produces a memory image of the STF. The STF can now be executed.

INTERROGATION COMMANDS

DISplays
Displays the contents of the objects

LISts
Lists the directory of the objects

FILE INTERFACE COMMANDS

DUMp
Passes results of current validation and outputs it to the host in a .DMP file.

USE
Takes command input from the specified file.

SPECIFICATIONS

Operating Environment

Intellec® Microcomputer Development System (Model 800, Series II, and Series III with 64K byte of RAM).

SUPPLIED EQUIPMENT
iSBC® 576—Speech Transaction Board with Speech Transaction Manager Firmware.
—Speech Transaction Generator software and firmware.
—Speech I/O Demo Software.
—iSBC® 575 Operator Control Unit.
—Shure SM-10A Microphone.
—Speech Transaction Design Manual.

OPTIONAL EQUIPMENT
iSBX®-351—RS232 Multimodule
iSBC®-342—EPROM expansion module
—SBX synthesizers

ORDERING INFORMATION

Part Number Description

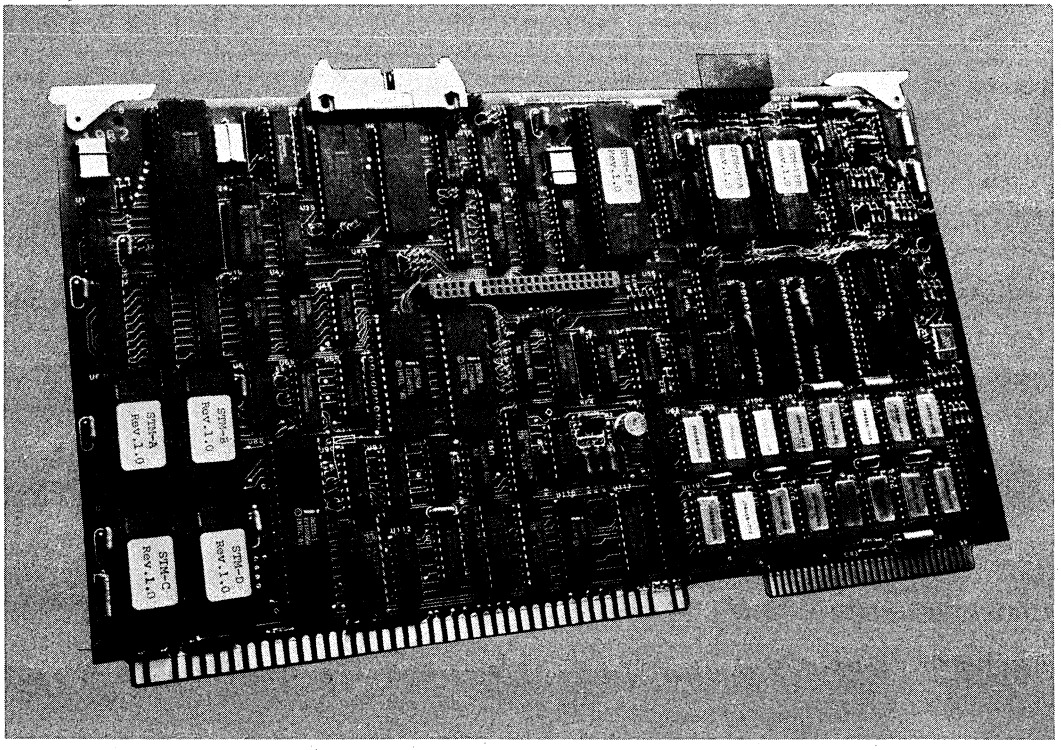
iSBC® 570 Speech Transaction Development Set

PRELIMINARY

iSBC® 576
SPEECH TRANSACTION BOARD

- **Up to 200 recognition words or phrases**
- **Automatic ASR and ESS handling**
- **On-board Speech Transaction Manager**
- **8086, 16-bit CPU**
- **On-board diagnostic**
- **Multibus or serial host interface**
- **iSBX® interface**
- **Built-in buffer editing functions**

The iSBC® 576 Speech Transaction Board is the heart of a speech I/O system. Beside providing Automatic Speech Recognition (ASR) capabilities, a ROM-resident Speech Transaction Manager (STM) is included on the board. This provides a flexibile operating structure for the system designer with a fully buffered speech-generated input-transaction handling capability. Flexibility has been designed into the STM to allow integration into existing applications without a major rewrite/redesign of host application software and hardware. The Speech Transaction Manager accommodates a Speech Transaction File which configures the iSBC® 576 Speech Transaction Board for each application. Also included on the board are three selectable audio feedback tones, visual feedback/control via a CRT terminal or printer, and an optional Electronic Speech Synthesis (ESS) capability.



FUNCTIONAL DESCRIPTION

Figure 6 shows the functional structure of the Speech Transaction Board.

Input Signal Conditioning—Microphone input signal is amplified and low-pass filtered. The conditioned signal is then digitized and passed through 16 bandpass digital filters implemented by 2920/21 analog signal processors. The 2920/21s are synchronized and are operating in parallel. The bandpass filter information is then assembled by an 8048 microcomputer for algorithm processing by an 8086 processor. System-to-system portability is guaranteed by the usage of digital signal processing techniques.

ASR—Automatic Speech Recognition is accomplished by the 8086 processor in conjunction with two 2920/21 digital signal processors and an 8048 microcomputer. ASR handling is done completely under the control of the Speech Transaction Manager. This task is transparent to the system designers. Automatic statistics are also provided to track system performance.

Tone Generator—3 audio tones are available for use as a prompt. The tones are generated within a 2920 analog signal processor. The tone generator also generates test patterns for use by the diagnostic section.

Diagnostic—Under the control of the Speech Transaction Manager, a diagnostic check of the speech recognition hardware and software can be performed. System integrity is automatically determined to insure repeatable performance.

Output Signal Conditioning—Output amplifiers are provided to drive a speaker for the audio tones. Volume can be varied by a potentiometer.

Terminal Driver—Under the control of the Speech Transaction Manager, a CRT terminal/keyboard can be connected directly to the Speech Transaction Board. The terminal can be used for visual feedback as well as data entry/control. The interface is RS232 compatible.

Operator Control—Two LED lights to indicate recognition status and an operator attention button are provided. These functions are programmable under the control of the Speech Transaction Manager.

Operator Reference Patterns—Speech patterns for recognition are normally contained in RAM. The patterns are downloaded from the host processor under the control of the Speech Transaction Manager. The operator reference patterns are also generated under the control of the Speech Transaction Manager.

Speech Transaction Manager—The Speech Transaction Manager is the heart of the Speech Transaction Board. The Speech Transaction Manager controls all of the functions within the board. This firmware is

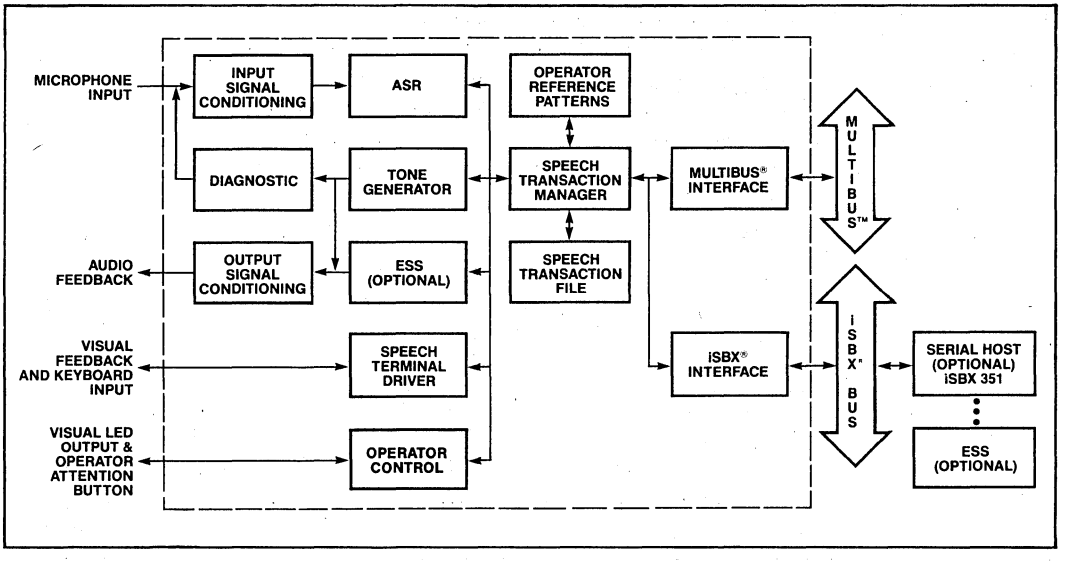


Figure 6. Functional Structure of the Speech Transaction Board

contained in 27128 EPROMs and is RMX®-88 (Real-Time Multi-Tasking Executive) based. Processing is provided by the 8086 processor.

Speech Transaction File—The Speech Transaction File determines the configuration of the board for each application. The Speech Transaction Manager executes this file which is normally downloaded from the host and-stored in RAM. The file can also be stored in ROM/EPROM on the Speech Transaction Board itself. These files are generated by the Speech Transaction Generator.

Multibus® Interface—A slave multibus® interface is implemented. On the multibus the Speech Transaction Board looks like a data port.

iSBX® Interface—One SBX® interface has been implemented. This interface is controlled by the Speech Transaction Manager. Interface with a non-Multibus® host can be implemented via this channel.

OPERATIONAL DESCRIPTION

The operation of the Speech Transaction Board is determined by the Speech Transaction Manager. The Speech Transaction Manager has several specific modes of operation as described below.

Speech Transaction Processing Mode—This mode enables the operator to enter by speech, or keyboard, a transaction message to a multibus or serial host.

File Mode—This mode supports file loading from the host through the multibus or serial interface. Loading and saving of operator reference patterns are also handled here.

Diagnostic Mode—This mode tests the hardware. The diagnostics will test the 2920/8048 interface and the 8048/8086 interface.

Terminal Mode—This mode provides for direct communication between the host and the Speech Transaction Board terminal. All response from the operator (through the terminal) is passed directly to the host. ALL host messages are passed directly to the terminal.

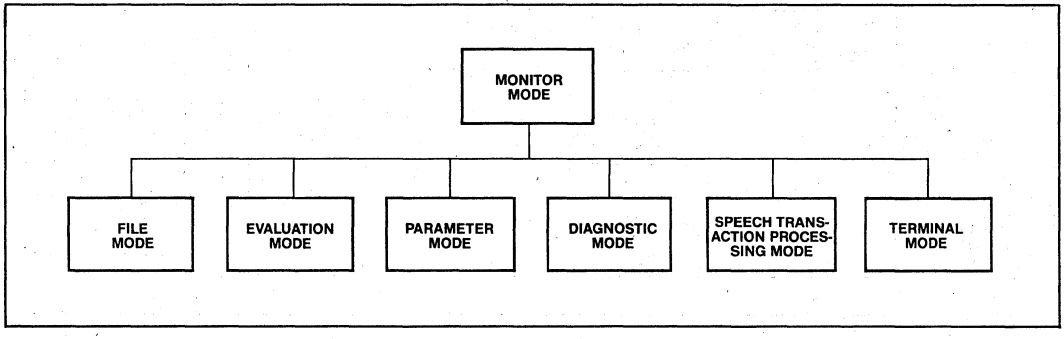
Parameter Mode—This mode lets the user define a limited set of configuration information and to set various other system parameters.

Evaluation Mode—This mode lets the user evaluate the recognition performance of an STF vocabulary or a vocabulary entered from the STB terminal. Use of this mode will facilitate evaluation of training strategies, vocabulary choices and parameter settings. In this mode statistics and automatic scoring of results are all standard features.

LIST OF COMMANDS

Monitor Mode Commands

STP—enter speech transaction processing mode
FIL—enter file mode
DIA—enter diagnostic mode
TER—enter terminal mode
PAR—enter parameter mode
MON—enter monitor mode
EVA—enter evaluation mode
HELP—list help commands
EXIT—exit current mode
INI—initialize statistics
RES—restores system status



Speech Transaction Processing Mode Function

Buffer Editing Functions

Forward
Backup
Correction
Replace
Forward Field
Backup Field
Erase Field
Continue
Beginning
Cancel
Finish

Utility Functions

Help—operator assistance at each field
Display—current transaction buffer
Next—go to next field
Detach—put terminal in "Terminal Mode"
Attach—get terminal out of "Terminal Mode"
Exit—exit STP mode
Up—raise rejection threshold
Down—lower rejection threshold
Relax—put system in not-ready state
Ready—first of two utterances to exit not-ready state
Attention—second of two utterances to exit not-ready state
Enable Transaction "N"—initiate transaction
Macro—performs a series of commands automatically in any mode

Operator Speech Pattern Maintenance Functions

Test Group
Test All
Retrain
Retrain Group
Retrain All
Delete
Delete All
Train
Train Group
Train All
Update
Update Group
Update All
Test

File Mode Commands

LST—load Speech Transaction File
SST—save speech transaction file
LRP—load operator speech patterns
SRP—save operator speech patterns
CRP—clear operator speech pattern RAM area
HELp—list help commands
CST—clear speech transaction
EXIt—exit current mode
LDI—load dictionary
SDI—save dictionary

Diagnostic Commands

FET—front end test
EXIt—exit mode
HELp—list help commands

Parameter Mode Commands

BLO—block size of transfer
CHS—communication header
CON—display all configuration parameters
DIS—discrimination level
DRE—small delta rejection
EST—display extended statistics
HOS—specifies host and characteristics
HTE—host terminator string
HTO—host time-out
INS—initialize statistics
MTP—minimum training passes
RPT—operator reference pattern names
SHC—serial host baud rate
STA—displays statistics
STF—STF name
STR—ROM STF name
TST—STB terminal status
WRD—word gap and word length
FEG—front-end gain
HELp—list help commands
EXIt—exit current mode

Evaluation Mode Commands

DEF—define
MVO—modify vocabulary
RVO—remove vocabulary
RRP—remove reference pattern
RET—retrain
LIS—list vocabulary
TRAin—train
UPDate—update
TESt—test
RECognition—recognition
STA—statistics
COR—cross correlation
INS—initialize statistics
HELp—list help commands
EXIt—exit current mode

SPECIFICATIONS

Operating Environment

Host Processor—any iSBC® Multibus® computer
—any RS232 serial host interface
Audio Input—475Ω input impedence
—50 m.v. p-p max.
—differential or single-ended

Equipment Supplied

iSBC® 576 Speech Transaction Board with Speech Transaction Manager Firmware

Optional Equipment

iSBX®-351	RS232 Multimodule
iSBX®-342	EPROM expansion SBX synthesizer
iSBC®-575	Operator Control Unit

Performance Specifications

Recognition vocabulary—200 words or phrases
Utterance duration—user selectable > 100 msec., minimum
—user selectable < 2 sec. maximum
Rejection Threshold—user selectable
Word gap—user selectable > 50 msec., minimum
—user selectable < 250 msec., maximum
Recognition Accuracy (50 state names)—99+%
Response Time (for vocabulary up to 200 words with maximum node length 50 words) — < 500 msec.

Physical Characteristics

Width—6.75 in. (17.15 cm)
Height—0.5 in. (1.27 cm)
Length—12.0 in. (30.48 cm)
Shipping weight—TBD
Mounting—occupies one slot of iSBC® system chasis in cardcage/backplane. With iSBX® Multimodule™ board mounted, vertical height increases to 1.13 in. (2.87 cm)

Electrical Characteristics

Power Requirements
+5V DC @ 3 A
+10V DC @ TBD *Multimodule™
−12V DC @ 0.02 A *Multimodule™
+12V DC @ 0.5 A

Environmental Characteristics

Temperature—0 to 55°C (operating): −55°C to 85°C (non-operating)
Humidity—up to 90% relative humidity without condensation (operating); all conditions without condensation or frost (non-operating)

Reference Manual

Speech Transaction Design Manual (supplied)

ORDERING INFORMATION

Part Number Description

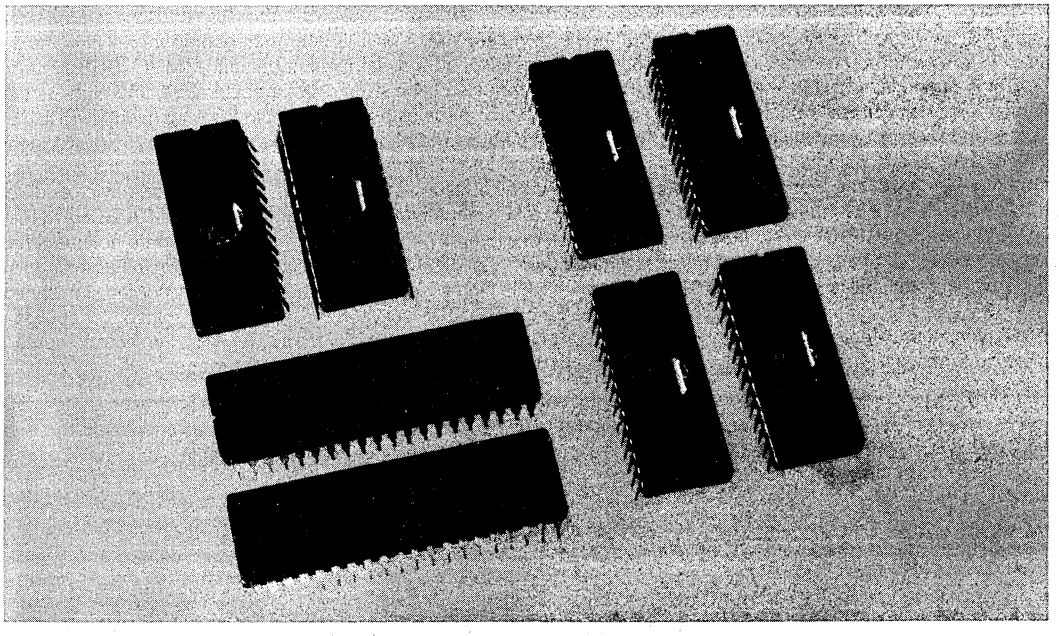
iSBC® 576	Speech Transaction Board

PRELIMINARY

iSBC® 577
SPEECH TRANSACTION RECOGNITION CHIP SET

- **High-volume solution for speech I/O**
- **Fully compatible with ISBC 570, 576-generated software**

The iSBC® 577 Speech Recognition Chip Set is a solution for your high-volume/maximum value-added speech I/O solution. The Chip Set contains the Intel-developed proprietary components from the iSBC® 576 Speech Transaction Board. With these components you can build the equivalent of the Speech Transaction Board into your own system. The Chip Set contains the digital front-end processors, a preprogrammed 8048 interface processor, the Speech Transaction Manager Firmware on 27128 EPROMs, and the 8086 microprocessor.



SPECIFICATIONS

Performance

—Refer to iSBC® 570 and iSBC® 576 performances.

Equipment Supplied

2—Preprogrammed 2920/21s (Digital Front-end Processor)
4—Preprogrammed 27128 (Speech Transaction Manager)
1—Preprogrammed 8048 (Interface Processor)
1—8086

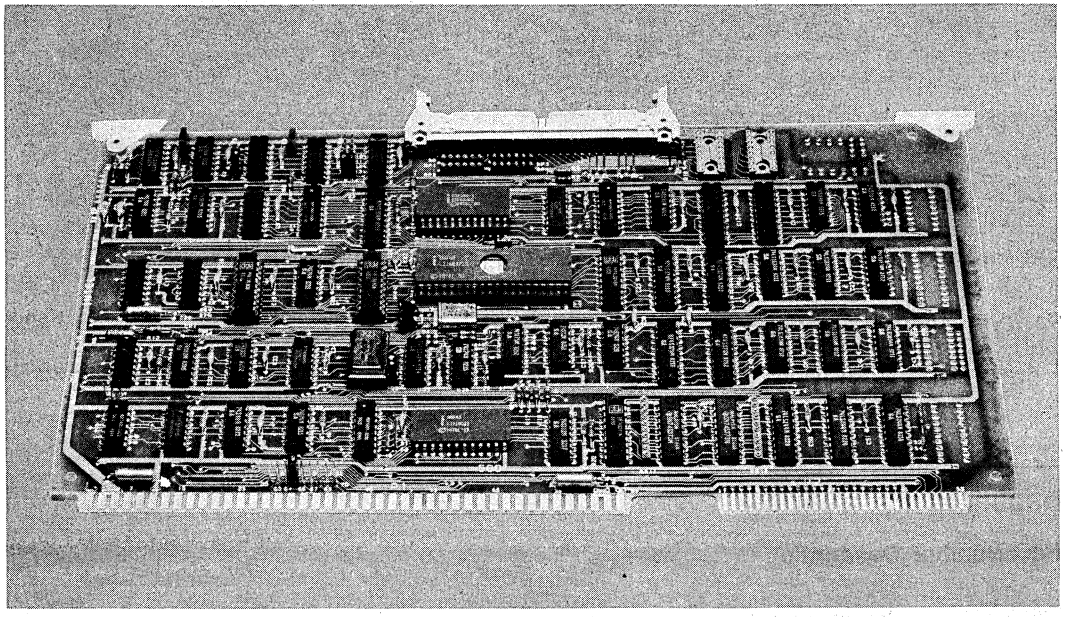
ORDERING INFORMATION

Part Number	Description
iSBC® 577	Speech Transaction Recognition Chip Set

iSBC® 580 MULTICHANNEL™ BUS TO iLBX™ BUS INTERFACE

- **MULTICHANNEL™ I/O bus 16-bit Talker/Listener interface**
- **iLBX™ bus master interface (primary or secondary)**
- **Supports MULTIBUS® interrupts**
- **Data rates up to 5.3 megabytes per second**
- **Addresses up to 16 megabytes of iLBX™ bus memory**
- **MULTIBUS® form factor**

The iSBC® 580 Interface Board is a member of Intel's complete line of MULTIBUS® microcomputers which maximize system performance by using separate optimized buses for intra-system communication (MULTIBUS system bus), high speed I/O (MULTICHANNEL™ DMA I/O bus), expansion I/O (iSBX™ I/O expansion bus) and high-speed memory expansion (iLBX™ execution bus). The iSBC 580 board provides a key element in the enhanced MULTIBUS system architecture by implementing a MULTICHANNEL I/O bus to iLBX bus interface on a single 6.75 x 12.00 inch printed circuit board. Using an LSI state machine with standard on-chip firmware to maximize throughput, the on-board Intel® 8048 Single Component Microcomputer transfers data between a MULTICHANNEL Controller, device and up to 16 megabytes of iLBX bus resident memory at rates up to 5.3 megabytes per second. Acting as a MULTICHANNEL Talker/Listener, the iSBC 580 board increases the system's overall performance by transferring data between the MULTICHANNEL I/O bus and system memory without using the MULTIBUS system bus. As shown in Figure 1, this allows other system tasks to utilize MULTIBUS resources while high-speed I/O block transfers are occurring simultaneously. The board's high throughput and independence from MULTIBUS activities make it an ideal solution for applications that must transfer large amounts of data in and out of a MULTIBUS system, such as MULTIBUS to host computer links and mass storage, graphics display and high-speed data acquisition subsystem interfaces.



FUNCTIONAL DESCRIPTION

MULTICHANNEL™ Interface Capabilities

The MULTICHANNEL I/O bus is designed to provide a general purpose, high-speed data path between a microcomputer system and up to 15 block transfer devices. Using a 16-bit wide data bus and a simple asynchronous handshaking scheme, the MULTICHANNEL bus can operate over distances up to 15 meters (50 feet) with a maximum burst throughput of 8 megabytes/second. The bus consists of 16 address/data lines, 6 control lines, 2 interrupt lines, parity lines and reset. Via these signals, a MULTICHANNEL Supervisor or Controller may configure and then initiate a block data transfer with any other device on the bus.

The iSBC 580 board acts as a 16-bit only Talker/Listener device on the MULTICHANNEL I/O bus. As a Talker/Listener, the board will respond to Register Read or Write and DMA requests issued by the MULTICHANNEL Supervisor (typically an iSBC 589 board) or by a MULTICHANNEL Controller device.

The iSBC 580 board implements 32 MULTICHANNEL Device Registers. The first three registers are the standard STO Status, SRQ Status and SRQ Mask Registers, as defined by the MULTICHANNEL Bus Specification. The remaining registers are used to communicate with the on-board firmware and for user data storage. The firmware operations which may be initiated by writing to the Command Register are listed in Table 1. The iSBC 580 board always sends and receives a 16-bit word on the MULTICHANNEL interface but, the iSBC® 580 device registers (see Table 2) are 8-bit only. Register Write operations use only the low order 8-bits (AD0-AD7). Register Read operations place the data on the low order data lines of the MULTICHANNEL I/O bus and set the high order data lines to FFH.

Command Code (Hex)	Operation
0	No Operation
1	Go off line forever
2	STO poll (diagnostic)
3	SRQ poll (diagnostic)
4	Set on-board timer
5	Read on-board timer
6	Start on-board timer
7	Stop on-board timer
8	Generate Task Complete interrupt
9	Perform checksum on firmware (diagnostic)
A	Turn on-board LED on
B	Turn on-board LED off
C	Reset
D, E	Reserved
F	Set interrupt mask
10	Read interrupt mask
11-1F	Reserved

Table 1. iSBC® 580 Firmware Commands

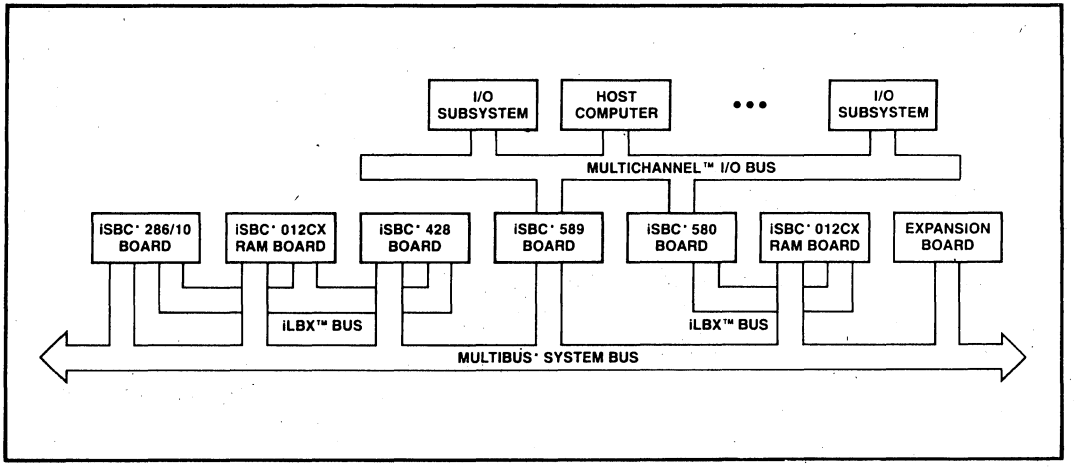


Figure 1. iSBC® 580 board, configured as an iLBX™ Bus Primary Master, transfers data between iLBX™ memory and MULTICHANNEL™ devices without using the system bus. The iSBC® 589 board acts as the MULTICHANNEL™ Supervisor and performs data transfers between MULTIBUS® memory and MULTICHANNEL™ devices.

The iSBC 580 board can generate maskable MULTICHANNEL STO interrupts when the board detects a parity error in incoming MULTICHANNEL data, when the board attempts to address non-existent iLBX memory or when the board detects a MULTIBUS interrupt from the system in which it resides. The last type of interrupt allows a single board computer to send an interrupt via the iSBC 580 board to the MULTICHANNEL Supervisor located in another MULTIBUS system. The board can also generate a number of SRQ interrupts on the MULTICHANNEL bus as shown in Figure 2.

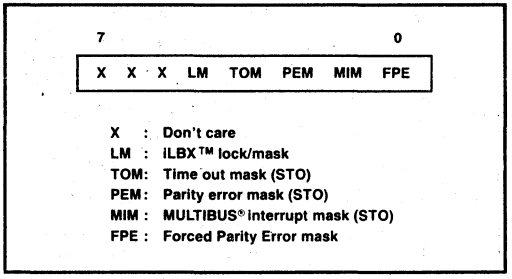


Figure 2. iSBC® 580 Interrupt Mask Register (14H)

iLBX™ Bus Interface Capabilities

Used in conjunction with the MULTIBUS interface, the iLBX bus is designed to provide off-board memory and I/O expansion for single board computers while maintaining on-board performance. The iLBX bus provides high-speed access to compatible expansion boards by granting privileged use of the bus to a single Primary Master. The bus also provides limited access to iLBX bus expansion boards for, at most, one Secondary Master that requires only occasional or non-concurrent access to iLBX resources. The iLBX bus, with 16 data lines, 24 address lines plus control, parity and interrupt signals, utilizes all the pins on the P2 connector except the four pins dedicated to the high-order address lines of the MULTIBUS interface. The non-multiplexed address and data lines provide access to up to 16 megabytes of iLBX bus resident memory, on up to 4 separate expansion boards, at speeds comparable to that of a single board computer's on-board resources.

The iSBC 580 board is configurable as either a Primary or a Secondary Master on the iLBX bus. Figure 1 shows a typical system configuration, with an iSBC 580 board acting as a Primary Master. The board can access up to 16 megabytes of iLBX memory. Supporting 16-bit transfers on the MULTICHANNEL bus, the board accesses memory as 16-bit words on even byte iLBX address boundaries. To increase the performance of iLBX memory read operations, the iSBC 580 board prefetches data from memory while the current data word is being transferred over the MULTICHANNEL I/O bus.

Register	Address
STO Status	00H
SRQ Status	01H
SRQ Mask	02H
RESERVED	03H-0FH
General Purpose Registers	10H*-1FH

* **NOTE:** 10H used as Command Register.

Table 2. iSBC® 580 MULTICHANNEL™ Device Register Set

SPECIFICATIONS

MULTICHANNEL™ Bus

Interface — Basic Talker/Listener

Transfer Mode — 16-bit

Device Address — Jumper selectable between 00H and 0EH

Registers — STO status, SRQ status, SRQ mask plus device specific registers

Signal Level — TTL compatible

iLBX™ Bus

Interface — Primary or Secondary (default) Master

Transfer Mode — 16-bit

Addressing — 16 megabytes on even byte boundaries only

Signal Level — TTL compatible

MULTIBUS® Interface

Data — None

Addressing — None

Interrupts — Jumper configurable to use any 1 of the 8 MULTIBUS interrupt lines. Interrupts are edge triggered.

Signal Level — TTL compatible

Throughput

5.3 megabytes/sec (2.65 megatransfers) max.

Connectors

iLBX™ BUS INTERFACE

Double-Sided Pins — 60

Centers — 0.100 in.

Mating Connectors* — Kelam RF30-2803-5
T&B Ansley A3020
(609-6025 modified)

MULTICHANNEL™ BUS INTERFACE

Pins — 60

Centers — 0.100 in.

Mating Connectors* — 3M 3334-6000
Berg 65949-960

* Connectors compatible with those listed may also be used.

Physical Characteristics

Width — 12.00 inches (30.5 cm)

Height — 6.75 inches (17.1 cm)

Depth — 0.60 inches (1.5 cm)

Weight — 12 ounces (340 gm)

Environmental Characteristics

Operating Temperature — 0° to 55°C

Relative Humidity — to 90% (without condensation)

DC Power Requirements

Voltage — +5 volt only ±5%

Current — 2.5 amps (typical)

Reference Manuals

144457-001 — iSBC 580 MULTICHANNEL to iLBX Bus Interface Board Hardware Reference Manual (NOT SUPPLIED)

143269-001 — Intel MULTICHANNEL Bus Specification (NOT SUPPLIED)

144456-001 — Intel iLBX Bus Specification (NOT SUPPLIED)

142996-001 — iSBC 589 Intelligent DMA Controller Board Hardware Reference Manual (NOT SUPPLIED)

Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, CA 95051

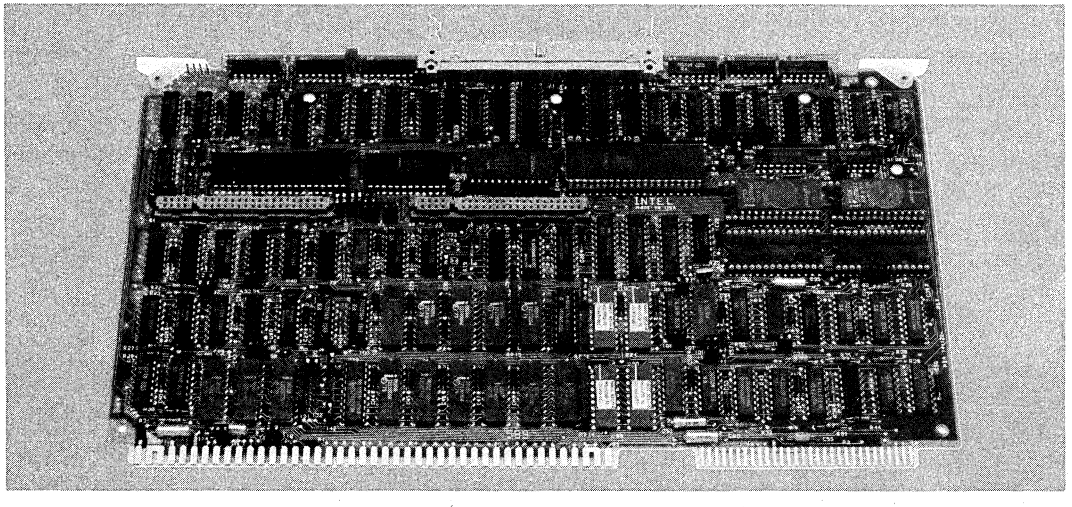
ORDERING INFORMATION

Part Number	Description
SBC 580	MULTICHANNEL to iLBX Bus Interface Board

iSBC® 589
INTELLIGENT DMA CONTROLLER

- **Configurable as either an intelligent slave or MULTIBUS® master**
- **5 MHz 8089 I/O Processor**
- **MULTICHANNEL™ DMA I/O bus interface with Supervisor, Controller or Basic Talker/Listener capabilities**
- **Two 8/16-bit iSBX™ bus connectors**
- **DMA transfer rates up to 1.25 megabytes per second**
- **User Command Interface Firmware Package provides high level I/O commands**
- **8K bytes of high-speed dual-ported static read/write memory**
- **Sockets for up to 32K bytes of read only memory or additional byte-wide static RAMs**
- **Three programmable timers**

The iSBC 589 Intelligent DMA Controller is a member of Intel's complete line of MULTIBUS microcompter systems which take full advantage of VLSI technology to provide economical computer based solutions for OEM applications. The iSBC 589 board is a general purpose, programmable, high-speed DMA controller on a single 6.75 × 12.00 inch printed circuit board. Using the board's dual-port RAM and standard EPROM resident firmware, the on-board Intel 8089 I/O Processor can perform memory to memory block transfers and complex I/O operations via two iSBX connectors and the MULTICHANNEL I/O bus at DMA transfer rates up to 1.25 megabytes per second. Acting as an intelligent slave to one or more iSBC 286, iSBC 186, iSBC 86, iSBC 88 or iSBC 80, single board computers, the iSBC 589 board enhances the sytem's overall performance by relieving the host CPU of time consuming I/O operations. The board's unique combination of performance, on-board intelligence and flexible hardware I/O interfaces make the iSBC 589 board the ideal solution for applications with specialized I/O requirements, such as high-speed data acquisition, graphics, instrument automation and specialized peripheral control, that previously would have necessitated an expensive custom designed I/O controller.



February, 1982
Order Number: 210354-001

FUNCTIONAL DESCRIPTION

Two Modes of Operation

The iSBC 589 Intelligent DMA Controller is capable of operating either as a stand-alone, high-speed data acquisition controller or as an intelligent slave. In stand-alone mode, external requests cause the Intel 8089 I/O Processor to execute I/O programs contained in its on-board memory. As an intelligent slave to one or more Intel single board computers, the IOP can perform sophisticated DMA operations in response to high level commands issued by the host processor. While operating in either mode, the iSBC 589 board may act as a MULTIBUS master to access any system memory or I/O resources.

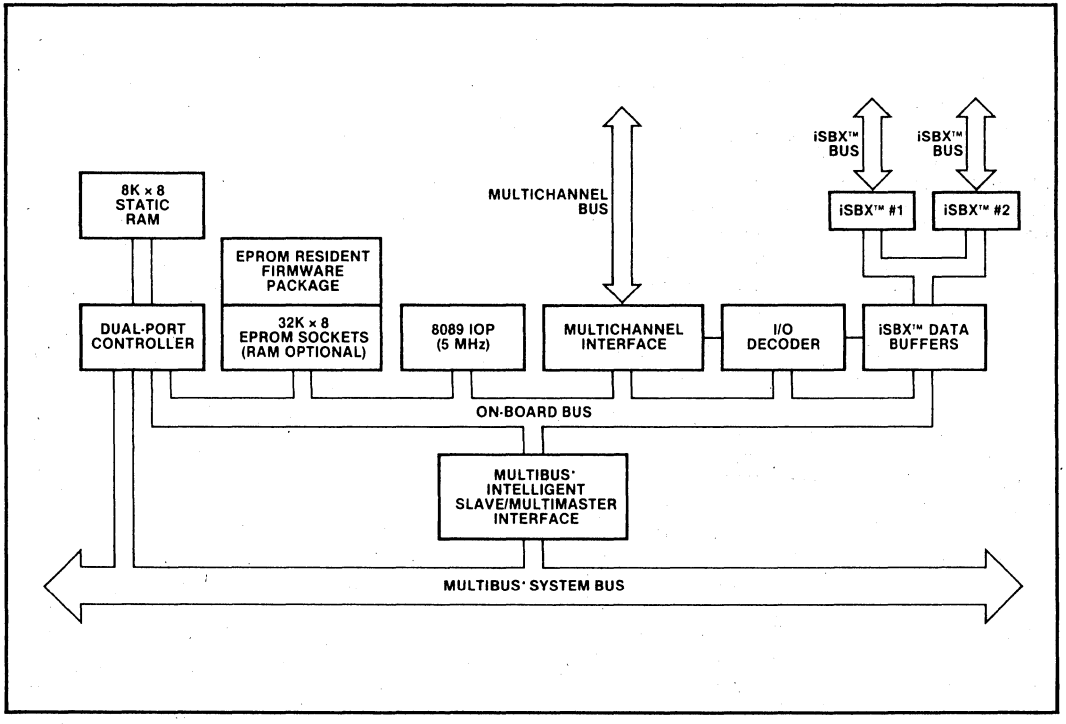
Input/Output Processor

The iSBC 589 board contains a 5 MHz Intel 8089 HMOS I/O Processor, whose architecture and instruction set have been optimized for performing DMA operations. The DMA function of the 8089 IOP uses a two cycle approach where the information actually flows through the 8089 IOP. This approach to DMA vastly simplifies the bus timings and enhances compatibility with memory and peripherals, in addition to allowing operations to be performed on the data as it is transferred. Operations can include such constructs as translate, where the 8089 automatically vectors through a lookup table and mask compare, both on the "fly". This DMA capability includes flexible termination conditions (such as external terminate, mask compare, single transfer and byte count expired).

The 8089 IOP supports two logically and physically separate I/O channels. The IOP maintains separate register sets for each I/O channel which allows the processor to alternate operation between the two channels without incurring context switching overhead delays.

DMA Capabilities

The iSBC 589 board supports both individual byte or word data transfers and DMA block transfer operations among its MULTICHANNEL interface, two iSBX connectors, on-board RAM and the MULTIBUS interface. Each of these devices may be combined



with any other as the source and destination for a DMA operation. The same firmware commands are used for all of the DMA source and destination combinations.

MULTICHANNEL Capabilities

The MULTICHANNEL bus provides a high-speed 8-bit or 16-bit wide data path for block data transfers between external devices, such as instruments, peripherals and other computers, and the iSBC 589 board. The iSBC 589 board can access up to 15 other devices on the MULTICHANNEL bus at distances of up to 15 meters and has the ability to address up to 16 megabytes of memory and 16 megabytes of I/O on each device.

The iSBC 589 Intelligent DMA Controller can interface to the MULTICHANNEL bus in one of three modes: as a Basic Talker/Listener, a Controller, and a Supervisor. In Basic Talker/Listener Mode, the iSBC 589 board monitors the MULTICHANNEL for requests from a Controller or the bus Supervisor to perform a read or a write operation, but it has no bus control capabilities. In Controller Mode, the board can request temporary control of the MULTICHANNEL bus from the bus Supervisor and thus initiate data transfer operations. In its MULTICHANNEL Supervisor configuration, the iSBC 589 has the capability to initiate data transfers on the bus, program other devices on the MULTICHANNEL bus, resolve and grant bus priority to other devices, monitor bus status, handle bus interrupts and control the MULTICHANNEL bus reset line. All of these functions are maintained by the on-board firmware based on parameter inputs from the host. Please refer to the MULTICHANNEL BUS SPECIFICATION for detailed descriptions of these modes.

iSBX™ Bus Capabilities

The iSBC 589 Controller contains two iSBX connectors which can support either 8-bit or 16-bit MULTIMODULE boards. The iSBX connectors are situated so that either two single-wide modules or one single-wide and one double-wide MULTIMODULE board may be installed. A wide variety of standard peripheral controllers and analog and digital I/O MULTIMODULE boards are currently available. In addition, the iSBX connectors provide an opportunity to add over 30 square inches of user designed hardware to the iSBC 589 board which can be used to implement specialized I/O interfaces. For more information on specific iSBX MULTIMODULE boards, consult the Intel OEM Microcomputer System Configuration Guide.

MULTIBUS® Capabilities

MULTIBUS system memory and I/O resources may be used as the source or the destination for an iSBC 589 board transfer operation. The iSBC 589 DMA Controller may also be used as a high-speed data mover to transfer blocks of data from one MULTIBUS system RAM area to another. MULTIBUS system memory may also be used to store Parameter Blocks to be executed by the on-board firmware package. The iSBC 589 board, acting as a MULTIBUS Master, can access up to 16 megabytes of MULTIBUS memory and up to 64K MULTIBUS I/O locations.

Two MULTIBUS transfer modes are available. Selection of the desired mode is done via the Parameter Block. Transfer rates of up to 900K bytes per second may be achieved in shared bus mode, where the iSBC 589 board requests access to the system bus for 1.4 microseconds to transfer one byte or word to or from memory. In BUSLOCK mode, the iSBC 589 is established as the sole master which may access the system bus for the duration of the block data transfer. In BUSLOCK mode, the iSBC 589 board can transfer up to one megabyte per second.

User Command Interface Firmware Package

The iSBC 589 board is supplied with a firmware package contained in two Intel 2732A EPROMs that greatly simplifies programming by providing a high level software interface to the on-board resources. In the majority of applications, the board may be programmed entirely via the firmware and without writing any 8089 IOP assembly language code. The firmware package supports the two channel operation of the 8089 IOP. Each channel has its own Parameter Block area containing the required information for independent channel operation.

To invoke an I/O operation, the user creates one or more Parameter Blocks in memory which describe the desired operation. The firmware, which consists of a series of 8089 IOP assembly language task programs, will interpret the Parameter Blocks to configure the board's interfaces or to perform byte, word or DMA block transfers. Each Parameter Block consists of a command byte, status byte, data source and destination pointers and

other information as shown in Table 1. Commands recognized by the firmware package are listed in Table 2. The Execute User Task command is of special interest because it allows the user to extend the capabilities of the iSBC 589 board by adding his own 8089 IOP assembly language routines to the firmware package, while retaining the structure and standard functions supplied by the firmware.

Table 1. User Command Interface Firmware Parameter Block Byte Format

Command Byte
Status Byte
Command Chaining Pointer
Command Chaining Pointer
Command Chaining Pointer
Command Chaining Pointer
Device Number
MULTICHANNEL Data Type
Memory Pointer or Register Number
Memory Pointer or Register Number
Memory Pointer or Data Storage Location
Memory Pointer or Data Storage Location
Device Number
MULTICHANNEL Data Type
Memory Pointer or Register Number
Memory Pointer or Register Number
Memory Pointer
Memory Pointer
Byte Counter
Byte Counter
Byte Counter

In addition to executing transfer operations, the firmware package executes an initialization sequence which prepares the 8089 IOP and the on-board RAM, EPROM and I/O resources for further firmware execution.

RAM Capabilities

In its standard configuration, the iSBC 589 board contains 8K bytes of high-speed, dual-ported static RAM. The first 256 bytes are dedicated for use by the on-board firmware. The remaining on-board RAM may be used for storing additional Parameter Blocks for the firmware or as a data buffer for I/O operations. This memory is always addressed by the 8089 IOP as locations 0000H to 1FFFH. However, for MULTIBUS accesses through the dual-port, the RAM base address may be configured on any 8K-byte boundary in the first megabyte page of the MULTIBUS memory space. Users may install additional on-board RAM by placing two byte-wide RAMs in the 28-pin JEDEC standard sockets. The additional RAM is accessible only by the on-board 8089 IOP.

EPROM Capabilities

The iSBC 589 board can be configured with up to 32K bytes of non-volatile read only memory. Four 28-pin sockets are provided for the use of Intel 2716, 2732 and 2764 EPROMs or byte-wide RAMs.

Table 2. User Command Interface Firmware Package Commands

Command	Description
NO-OP	Test the intelligent slave interface on the iSBC 589 board. The board reads the Parameter Block, generates status and interrupts the host on completion.
REGISTER WRITE	Write either a word or byte of data from the Data Storage Location within the Parameter Block to the location specified by the Parameter Block Device Number and Register Number.
REGISTER READ	Read either a word or byte of data from the location specified by the Parameter Block Device Number and Register Number to the Data Storage Location within the Parameter Block.
PERFORM DMA	Transfer data beginning at the location specified by the source Memory Pointer, Device Number and Register Number parameters to the location specified by the destination Memory Pointer, Device Number and Register Number parameters. The number of transfers is specified by the Byte Count parameter. A Byte Count of 0 enables DMA until an external terminate condition is sensed.
EXECUTE USER TASK	Transfer 8089 IOP program execution from the Firmware Package to a user defined 8089 assembly language routine beginning at the location specified by the Memory Pointer parameter. Upon completion, the user task returns control to the firmware.

In the default configuration, the board is jumpered for 32K devices, and, two 2732A EPROMs containing the firmware package are installed. Users who wish to extend the capabilities of the firmware may do so by programming unused locations in the firmware PROMs, installing two additional 2732A PROMs or copying the firmware into 2764s along with their own code. As an alternative, two byte-wide RAMs of equal or smaller capacity may be installed in the open sockets and used in conjunction with the firmware PROMs.

Programmable Interval Timers

Three independent, fully programmable 16-bit interval/event counters are provided by an 8254-12 Programmable Interrupt Timer. Each counter may operate in either BCD or binary mode. One counter is used by the firmware package, leaving two counters available to the firmware user. These timers may be used for a variety of on-board and off-board functions including timed-interval DMA requests and terminations or fail safe time out control for I/O operations.

System Development Capabilities

For applications where it is necessary to extend the User Command Firmware Package by writing additional 8089 IOP assembly language code, the development cycle can be significantly reduced and simplified by using the Intellec Series Microcomputer Development Systems. The 8089 IOP Software Support Package which includes a Macro assembler, linker, locater and PROM mapper is supported by the ISIS-II disk-based operating system.

In-Circuit Emulator

The ICE-86A or ICE-86 and ICE-86U upgrade kit provide the necessary link between the software development environment provided by the Intellec system and the "target" iSBC 589 execution system. In addition to providing a mechanism for loading executable code and data into the iSBC 589 board, the In-Circuit Emulator provides a sophisticated command set to assist in debugging software and in final integration of the user hardware and software.

SPECIFICATIONS

8089 IOP

WORD SIZE

Instruction — 16 to 40-bits

Data — 8, 16-bits

SYSTEM CLOCK

5.0 MHz ± 0.1%

CYCLE TIME

2.2 microseconds for the fastest instructions

System Access Time

Dual-port Memory — 550 nanoseconds (worst case, without contention from on-board access)

I/O Capacity

MULTICHANNEL I/O Bus — 1 MULTICHANNEL port which supports 8 and 16-bit transfers and can be configured as a Basic Talker/Listener, Controller or Supervisor

iSBX™ MULTIMODULE™ — Two (2) iSBX MULTIMODULE boards

I/O Addressing

Interface	I/O Addresses
iSBX Connector #1	FF80 thru FF9F
iSBX Connector #2	FFA0 thru FFBF
MULTICHANNEL	FFD0 thru FFEE
Interval Timer	FFC8 thru FFCE
Other On-board Devices	FFC0 thru FFC6 FFF0 thru FFFE

Memory Capacity

ON-BOARD EPROM

Device	Total Capacity	Address Range
2716	8K bytes	$FE000\text{-}FFFFF_H$
2732A	16K bytes	$FC000\text{-}FFFFF_H$
2764	32K bytes	$F8000\text{-}FFFFF_H$

ON-BOARD RAM

Total Capacity — 8K bytes

On-Board Address — $00000\text{-}01FFF_H$

MULTIBUS® Address — Jumper selectable on 8K byte boundaries. Default is 0_H.

I/O Transfer Rates (microseconds/tranfer)

	MULTICHANNEL	iSBX™	MULTIBUS®		On-Board RAM
			Shared	Buslock	
MULTICHANNEL	—	2.0	2.4	2.2	1.8
iSBX	2.0	2.0	2.4	2.2	2.0
MULTIBUS (Shared)	2.4	2.4	2.8	—	2.2
MULTIBUS (Buslock)	2.2	2.2	—	2.4	2.0
On-Board RAM	1.8	1.8	2.2	2.0	1.6

Timers

Input Frequencies — Jumper selectable at 1.25 MHz, 625 KHz or 312.5 KHz

Output Frequencies/Timing Intervals —

Function	Single Timer/Counter		Dual Timer/Counter (Two Timers Cascaded)	
	Minimum	Maximum	Minimum	Maximum
Real-time delay	1.6 usec	210 msec	3.2 usec	1.37×10^4 sec
Programmable one-shot	1.6 usec	210 msec	3.2 usec	1.37×10^4 sec
Rate generator	4.76 Hz	625 KHz	7.3×10^{-5} Hz	312.5 KHz
Square-wave rate generator	4.76 Hz	625 KHz	7.3×10^{-5} Hz	312.5 KHz
Software triggered strobe	1.6 usec	210 msec	3.2 usec	1.37×10^4 sec
Hardware triggered strobe	1.6 usec	210 msec	3.2 usec	1.37×10^4 sec

Connectors

Interface	Double-Sided Pins (qty.)	Centers (in.)	Mating Connectors*
MULTIBUS System Bus	86	0.156	ELFAB BS1562043PBB Viking 2KH43/9AMK12 Soldered PCB Mount EDAC 337086540201 ELFAB BW1562D43PBB EDAC 337086540202 ELFAB BW1562A43PBB Wire Wrap
Auxiliary Bus	60	0.100	EDAC 345060524802 ELFAB BS1020A30PBB EDAC 345060540201 ELFAB BW1020D30PBB Wire Wrap
iSBX Bus (2)	36	0.100	iSBX 960-5
MULTICHANNEL Bus	60	0.100	3M 3334-6000 BERG 65949-960

*NOTE: Connectors compatible with those listed may also be used.

Interfaces

MULTIBUS® — All signals TTL compatible

MULTICHANNEL — All signals TTL compatible

iSBX™ Bus — All signals TTL compatible

Timers — All signals TTL compatible

Auxiliary Power/Memory Protect

There is no provision made on the iSBC 589 board for battery backup of RAM or for power fail detection.

MULTIBUS® Bus Drivers

Function	Characteristic	Sink Current (mA)
Data	Tri-state	32
Address	Tri-state	32
Commands	Tri-state	32

Physical Characteristics

Width — 12.00 in (30.48 cm)

Height — 7.05 in (17.9 cm)

Depth — .50 in (1.27 cm)

Weight — 16 oz (453.6 gm)

Environmental Characteristics

Operating Temperature — 0°C to 55°C

Relative Humidity — to 90% (without condensation)

Electrical Characteristics

DC POWER REQUIREMENTS

Configuration	Current Requirements (+5V + 5% maximum)
Without EPROM	4.7 amps
With 8K EPROM (using four 2716s)	5.4 amps
With 8K EPROM* (using two 2732As)	5.0 amps
With 16K EPROM (using four 2732As)	5.3 amps
With 32K EPROM (using four 2164s)	5.3 amps

* Factory default configuration

Reference Manuals

142996-001 — iSBC 589 Intelligent DMA Controller Board Hardware Reference Manual (Not Supplied)

142686-001 — Intel iSBX Bus Specification (Not Supplied)

143269-001 — Intel MULTICHANNEL Bus Specification (Not Supplied)

Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051

ORDERING INFORMATION

Part Number	Description
SBC 589	Intelligent DMA Controller Board

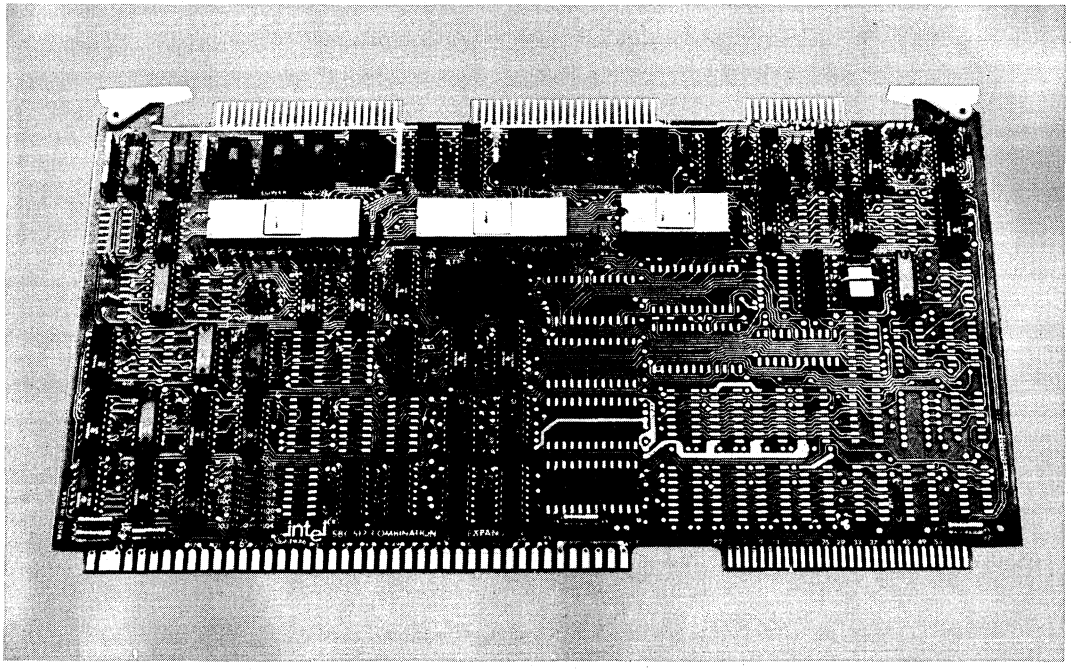
Digital I/O Expansion and Signal Conditioning Boards

11

iSBC® 517
COMBINATION I/O EXPANSION BOARD

- **48 programmable I/O lines with sockets for interchangeable line drivers and terminators**
- **Synchronous/asynchronous communications interface with RS232C drivers and receivers**
- **Eight maskable interrupt request lines with a pending interrupt register**
- **1 ms interval timer**

The iSBC 517 Combination I/O Expansion Board is a member of Intel's complete line of iSBC memory and I/O expansion boards. The board interfaces directly with any iSBC single board computer via the system bus to expand serial and parallel I/O capacity. The combination I/O board contains 48 programmable parallel I/O lines. The system software is used to configure the I/O lines to meet a wide variety of system peripheral requirements. The flexibility of the I/O interface is significantly enhanced by the capability of selecting the appropriate combination of optional line drivers and terminators to provide the required sink current, polarity, and drive/termination characteristics for each application. A programmable RS232C communications interface is provided on the iSBC 517. This interface may be programmed by the system software to provide virtually any asynchronous or synchronous serial data transmission technique (including IBM Bi-Sync). A comprehensive RS232C interface to CRTs, RS232C compatible cassettes, and asynchronous and synchronous modems is thus on the board. An on-board register contains the status of eight interrupt request lines which may be interrogated from the system bus, and each interrupt request line is maskable under program control. The iSBC 517 also contains a jumper selectable 1 ms interval timer and interface logic for eight interrupt request lines.



FUNCTIONAL DESCRIPTION

Programming Flexibility

The 48 programmable I/O lines on the iSBC 517 are implemented utilizing two Intel 8255 programmable peripheral interfaces. The system software is used to configure these programmable I/O lines in any of the combinations of unidirectional input/output, and bi-directional ports indicated in Table 1. In order to take full advantage of the large number of possible I/O configurations, sockets are provided for interchangeable I/O line drivers and terminators to provide the required sink current, polarity, and drive/termination characteristics for each application. The 48 programmable I/O lines and signal ground lines are brought out to two 50-pin edge connectors that mate with flat, round, or woven cable. Typical I/O read access time is 280 nanoseconds. Typical I/O read cycle time is 600 nanoseconds.

Communications Interface

The programmable communications interface on the iSBC 517 is provided by an Intel 8251 Universal Synchronous/Asynchronous Receiver/Transmitter (USART). The USART can be programmed by the system software to select the desired asynchronous or synchronous serial data transmission technique (including IBM Bi-Sync). The mode of operation (i.e., synchronous or asynchronous), data format, control character format, parity, and asynchronous serial transmission rate are all under program control. The 8251 provides full duplex, double-buffered transmit and receive capability, and parity, overrun, and framing error detection are all incorporated in the USART. The comprehensive RS232C interface on the board provides a direct interface to RS232C compatible equipment. The RS232C serial data lines and signal ground lines are brought out to a 26-pin edge connector that mates with RS232C compatible flat or round cables.

Interrupt Request Lines

Interrupt requests may originate from eight sources. Four jumper selectable interrupt requests can be automatically generated by the programmable peripheral interface when a byte of information is ready to be transferred to the CPU (i.e., input buffer is full) or a character has been transmitted (i.e., output data buffer is empty). Two jumper selectable interrupt requests can be automatically generated by the USART when a character is ready to be transferred to the CPU (i.e., receive buffer is full) or a character has been transmitted (transmit buffer is empty). These six interrupt request lines are all maskable under program control. Two interrupt request lines may be interfaced directly from user designated peripheral devices via the I/O edge connector. An on-board register contains the status of all eight interrupt request lines, and may be interrogated by the CPU. Each interrupt request line is

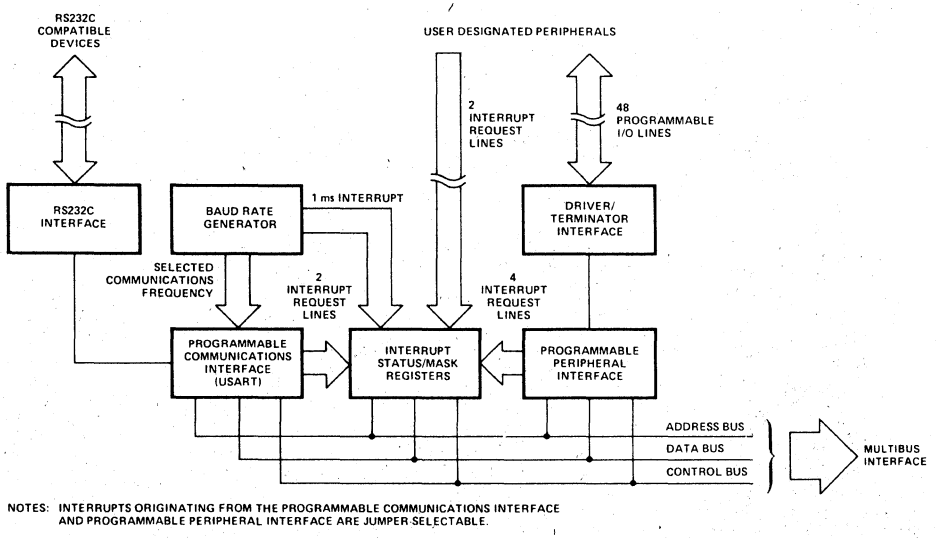


Figure 1. iSBC ® 517 Combination I/O Expansion Board Block Diagram

Table 1. Input/Output Port Modes of Operation

Ports	Lines (qty)	Mode of Operation					
		Unidirectional				Bidirectional	Control
		Input		Output			
		Unlatched	Latched & Strobed	Latched	Latched & Strobed		
1	8	X	X	X	X	X	
2	8	X	X	X	X		
3	4	X		X			X[1]
	4	X		X			X[1]
4	8	X	X	X	X	X	
5	8	X	X	X	X		
6	4	X		X			X[2]
	4	X		X			X[2]

Notes

1. Part of port 3 must be used as control port when either port 1 or port 2 are used as a latched and strobed input or a latched and strobed output port or port 1 is used as a bidirectional port.
2. Part of port 6 must be used as a control port when either port 4 or port 5 are used as a latched and strobed input or a latched and strobed output port or port 4 is used as a bidirectional port.

maskable under program control. Routing for the eight interrupt request lines is jumper selectable. They may be ORed to provide a single interrupt request line for the iSBC 80/10B, or they may be individually provided to the system bus for use by other iSBC single board computers.

Interval Timer

Each board contains a jumper selectable 1 ms interval timer. The timer is enabled by jumpering one of the interrupt request lines from the I/O edge connector to a 1 ms interval interrupt request signal originating from the baud rate generator.

SPECIFICATIONS

I/O Addressing

Port	1	2	3	4	5	6	8255 No. 1 Control	8255 No. 2 Control	USART Data	USART Control
Address	X4	X5	X6	X8	X9	XA	X7	XB	XC	XD

Note

X is any hex digit assigned by jumper selection.

I/O Transfer Rate

Parallel — Read or write cycle time 760 ns max
Serial — (USART)

Frequency (kHz) (Jumper Selectable)	Baud Rate (Hz)		
	Synchronous	Asynchronous (Program Selectable)	
		÷16	÷64
153.6	—	9600	2400
76.8	—	4800	1200
38.4	38400	2400	600
19.2	19200	1200	300
9.6	9600	600	150
4.8	4800	300	75
6.98	6980	—	110

Serial Communications Characteristics

Synchronous — 5-8 bit characters; internal or external character synchronization; automatic sync insertion.

Asynchronous — 5-8 bit characters; peak characters generation; 1, 1½, or 2 stop bits; false start bit detectors.

Interrupts

Eight interrupt request lines may originate from the programmable peripheral interface (4 lines), the USART (2 lines), or user specified devices via the I/O edge connector (2 lines) or interval timer.

Interrupt Register Address

X1 Interrupt mask register
X0 Interrupt status register

Note

X is any hex digit assigned by jumber selection.

Timer Interval

1.003 ms ±0.1% when 110 baud rate is selected
1.042 ms ±0.1% for all other baud rates

Interfaces

Bus — All signals TTL compatible
Parallel I/O — All signals TTL compatible
Serial I/O — RS232C
Interrupt Requests — All TTL compatible

Connectors

Interface	Pins (qty)	Centers (in.)	Mating Connectors
Bus	86	0.156	CDC VPB01E43A00A1
Parallel I/O	50	0.1	3M 3415-000 or TI H312125
Serial I/O	26	0.1	3M 3462-000 or TI H312113
Auxiliary[1]	60	0.1	AMP PE5-14559 or TI H311130

Note

1. Connector heights and wire-wrap pin lengths are not guaranteed to conform to Intel OEM or system packaging. Auxiliary connector is used for test purposes only.

Line Drivers and Terminators

I/O Drivers — The following line drivers and terminators are compatible with all the I/O driver sockets on the iSBC 517:

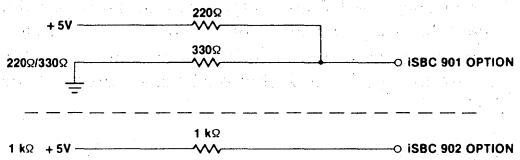
Driver	Characteristic	Sink Current (mA)
7438	I,OC	48
7437	I	48
7432	NI	16
7426	I,OC	16
7409	NI,OC	16
7408	NI	16
7403	I,OC	16
7400	I	16

Note

I = inverting; NI = non-inverting; OC = open-collector.

Ports 1 and 4 have 25 mA totem-pole drivers and 1 kΩ terminators.

I/O Terminators — 220Ω/330Ω divider or 1 kΩ pullup



Bus Drivers

Function	Characteristic	Sink Current (mA)
Data	Tri-state	50
Commands	Tri-state	25

Physical Characteristics

Width — 12.00 in. (30.48 cm)
Height — 6.75 in. (17.15 cm)
Depth — 0.50 in. (1.27 cm)
Weight — 14 oz (397. 3 gm)

Electrical Characteristics

Average DC Current

V_{CC} = +5V ±5%
V_{DD} = +12V ±5%
V_{AA} = −12V ±5%
I_{CC} = 2.4 mA max
I_{DD} = 40 mA max
I_{AA} = 60 mA max

Note

Does not include power required for optional I/O drivers and I/O terminators. With eight 220Ω/330Ω input terminators installed, all terminator inputs low.

Environmental Characteristics

Operating Temperature — 0°C to +55°C

Reference Manual

9800388B — iSBC 517 Hardware Reference manual (NOT SUPPLIED)

Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

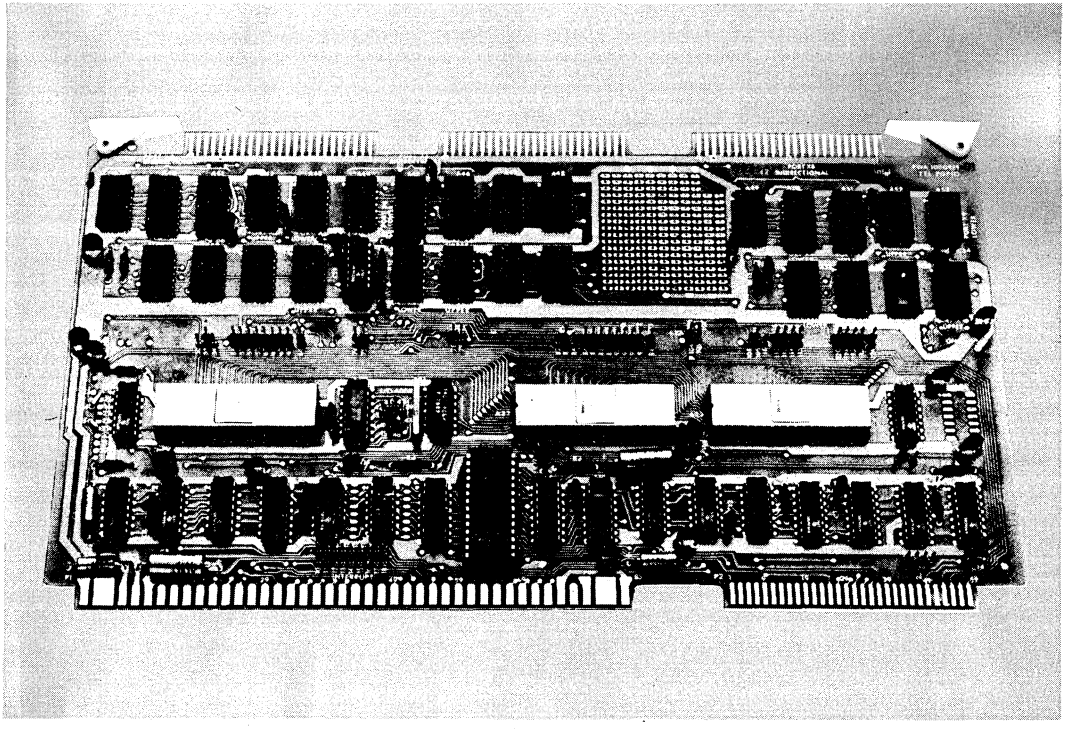
ORDERING INFORMATION

Part Number	Description
SBC 517	Combination I/O Expansion Board

iSBC® 519 (or pSBC 519*) PROGRAMMABLE I/O EXPANSION BOARD

- **iSBC® I/O expansion via direct MULTIBUS® Interface**
- **72 programmable I/O lines with sockets for interchangeable line drivers and terminators**
- **Jumper selectable I/O port addresses**
- **Jumper selectable 0.5, 1.0, 2.0, or 4.0 ms interval timer**
- **Eight maskable interrupt request lines with priority encoded and programmable interrupt algorithms**

The iSBC 519 Programmable I/O Expansion Board is a member of Intel's complete line of iSBC memory and I/O expansion boards. The iSBC 519 interfaces directly to any iSBC single board computer via the system bus to expand input and output port capacity. The iSBC 519 provides 72 programmable I/O lines. The system software is used to configure the I/O lines to meet a wide variety of peripheral requirements. The flexibility of the I/O interface is further enhanced by the capability of selecting the appropriate combination of optional line drivers and terminators to provide the required sink current, polarity, and drive/termination characteristics for each application. Address selection is accomplished by using wire-wrap jumpers to select one of 16 unique base addresses for the input and output ports. The board operates with a single +5V power supply.



*Same product, manufactured by Intel Puerto Rico, Inc.

FUNCTIONAL DESCRIPTION

The 72 programmable I/O lines on the iSBC 519 are implemented utilizing three Intel 8255A programmable peripheral interfaces. The system software is used to configure the I/O lines in any combination of unidirectional input/output and bidirectional ports indicated in Table 1. In order to take full advantage of the large number of possible I/O configurations, sockets are provided for interchangeable I/O line drivers and terminators. The 72 programmable I/O lines and signal ground lines are brought out to three 50-pin edge connectors that mate with flat, round, or woven cable.

Interval Timer

Typical I/O read access time is 350 nanoseconds. Typical I/O read/write cycle time is 450 nanoseconds. The interval timer provided on the iSBC 519 may be used to generate real time clocking in systems requiring the periodic monitoring of I/O functions. The time interval is derived from the constant clock (BUS CCLK) and the timing interval is jumper selectable. Intervals of 0.5, 1.0, 2.0, and 4.0 milliseconds may be selected when an iSBC single board computer is used to generate the clock. Other timing intervals may be generated if the user provides a separate constant clock reference in the system.

Eight-Level Vectored Interrupt

An Intel 8259A programmable interrupt controller (PIC) provides vectoring for eight interrupt levels. As shown in Table 2, a selection of three priority processing algorithms is available to the system designer so that the

Table 1. Input/Output Port Modes of Operation

Ports	Lines (qty)	Mode of Operation					Control
		Unidirectional				Bidirectional	
		Input		Output			
		Unlatched	Latched & Strobed	Latched	Latched & Strobed		
1,4,7	8	X	X	X	X	X	
2,5,8	8	X	X	X	X		
3,6,9	4	X		X			X[1,2,3]
	4	X		X			X[1,2,3]

Notes

1. Part of port 3 must be used as a control port when either port 1 or port 2 are used as a latched and strobed input or a latched and strobed output port or port 1 is used as a bidirectional port.
2. Part of port 6 must be used as a control port when either port 4 or port 5 are used as a latched and strobed input or a latched and strobed output port or port 4 is used as a bidirectional port.
3. Part of port 9 must be used as a control port when either port 7 or port 8 are used as a latched and strobed input or a latched and strobed output port or port 7 is used as a bidirectional port.

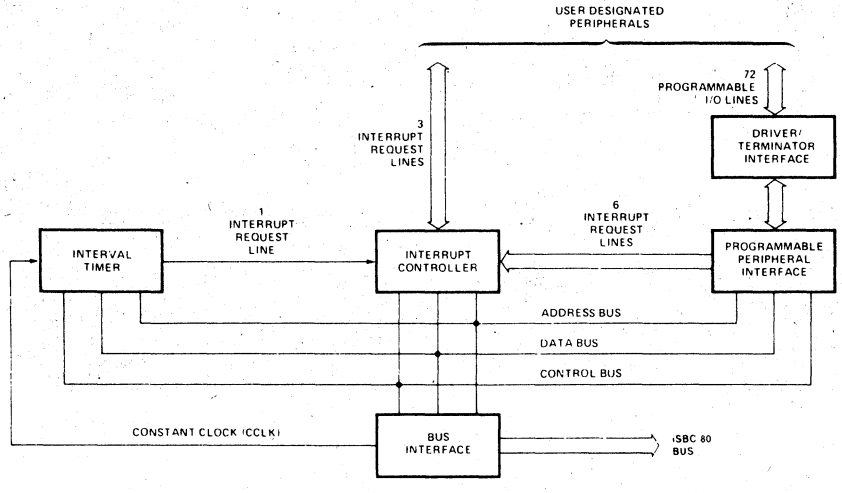


Figure 1. iSBC ® 519 Programmable I/O Expansion Board Block Diagram

Table 2. Interrupt Priority Options

Algorithm	Operation
Fully nested	Interrupt request line priorities fixed at 0 as highest, 7 as lowest.
Auto-rotating	Equal priority. Each level, after receiving service, becomes the lowest priority level until next interrupt occurs.
Specific priority	System software assigns lowest priority level. Priority of all other levels are based in sequence numerically on this assignment.

manner in which requests are serviced may be configured to match system requirements. Priority assignments may be reconfigured dynamically via software at any time during system operation. The PIC accepts interrupt requests from the programmable parallel I/O interfaces, the interval timer, or direct from peripheral equipment. The PIC then determines which of the incoming requests is of the highest priority, determines whether this request is of higher priority than the level currently being serviced, and if appropriate, issues an interrupt to the system master. Any combination of interrupt levels may be masked through storage, via software, of a single byte to the interrupt mask register of the PIC.

Interrupt Request Generation — Interrupt requests may originate from 10 sources. Six jumper selectable interrupt requests can be automatically generated by the programmable peripheral interfaces when a byte of information is ready to be transferred to the system master (i.e., input buffer is full) or a character has been transmitted (i.e., output data buffer is empty). Three interrupt request lines may be interfaced to the PIC directly from user designated peripheral devices via the I/O edge connectors. One interrupt request may be generated by the interval timer.

Bus Line Drivers — The PIC interrupt request output line may be jumper selected to drive any of the nine interrupt lines on the MULTIBUS. Any of the on-board request lines may also drive any interface interrupt line directly via jumpers and buffers on the board.

SPECIFICATIONS

Addressing

Port	1	2	3	8255 No. 1 Control	4	5	6	8255 No. 2 Control	7	8	9	8255 No. 3 Control
Address	X0	X1	X2	X3	X4	X5	X6	X7	X8	X9	XA	XB

Interrupts

Register Addresses (hex notation, I/O address space)

XD Interrupt request register
XC In-service register
XD Mask register
XC Command register
XD Block address register
XC Status (polling register)

Note
Several registers have the same physical address; sequence of access and one data bit of control word determines which register will respond.

Ten interrupt request lines may originate from the programmable peripheral interface (6 lines), or user specified devices via the I/O edge connector (3 lines), or interval timer (1 line).

Interval Timer

Output Register — Timer interrupt register output is cleared by an output instruction to I/O address XE or XF[1].

Timing Intervals — 500, 1,000, 2,000, and 4,000 ms ±1%; jumper selectable[2].

Notes
1. X is any hex digit assigned by jumper selection.
2. Assumes constant clock (CCLK) frequency of 9.216 MHz ±1%.

Interfaces

Bus — All signals TTL compatible
Parallel I/O — All signals TTL compatible
Interrupt Requests — All TTL compatible

Connectors

Interface	Pins (qty)	Centers (in.)	Mating Connectors
Bus	86	0.156	Viking 3KH43/9AMK12
Parallel I/O	50	0.1	3M 3415-000 or TI H312125
Serial I/O	26	0.1	3M 3462-000 or TI H312113
Auxiliary[1]	60	0.1	AMP PE5-14559 or TI H311130

Note
1. Connector heights and wirewrap pin lengths are not guaranteed to conform to Intel OEM or System packaging.

Line Drivers and Terminators

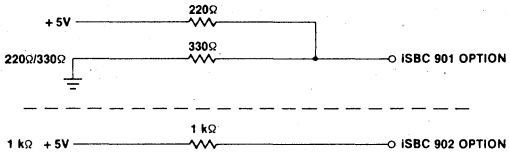
I/O Drivers — The following line drivers and terminators are compatible with all the I/O driver sockets on the iSBC 519:

Driver	Characteristic	Sink Current (mA)
7438	I,OC	48
7437	I	48
7432	NI	16
7426	I,OC	16
7409	NI,OC	16
7408	NI	16
7403	I,OC	16
7400	I	16

Note
I = inverting; NI = non-inverting; OC = open-collector.

AFN-00278A

I/O Terminators — 220Ω/330Ω divider or 1 kΩ pullup



Ports 1, 4, and 7 may use any of the drivers or terminators shown above for unidirectional (input or output) port configurations. Either terminator and the following bidirectional drivers and terminators may be used for ports 1, 4, and 7 when these ports are used as bidirectional ports.

Bidirectional Drivers

Driver	Characteristic	Sink Current (mA)
Intel 8216	NI, TS	25
Intel 8226	I, TS	50

Note

I = inverting; NI = non-inverting; TS = three-state.

Terminators (for ports 1, 4, and 7 when used as bidirectional ports)

Supplier	Product Series
CTS	760-
Dale	LDP14k-02
Beckman	899-1

Bus Drivers

Function	Characteristic	Sink Current (mA)
Data	Tri-state	50
Commands	Tri-state	25

Physical Characteristics

Width — 12.00 in. (30.48 cm)
Height — 6.75 in. (17.15 cm)
Depth — 0.50 in. (1.27 cm)
Weight — 14 oz (397.3 gm)

Electrical Characteristics

Average DC Current

Voltage	Without Termination[1]	With Termination[2]
V_{CC} = +5V ±5%	I_{CC} = 1.5A max	3.5A max

Note

1. Does not include power required for optional I/O drivers and I/O terminators.
2. With 18 220Ω/330Ω Input terminators installed, all terminator inputs low.

Environmental Characteristics

Operating Temperature — 0°C to +55°C

Reference Manual

9800385B — iSBC 519 Hardware Reference manual (NOT SUPPLIED)

Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

ORDERING INFORMATION

Part Number	Description
SBC 519	Programmable I/O Expansion Board

iSBC®556
OPTICALLY ISOLATED I/O BOARD

- **Up to 48 digital optically isolated input/output data lines for MULTIBUS® systems**
- **Choice of**
  - **— 24 fixed input lines**
  - **— 16 fixed output lines**
  - **— 8 programmable lines**
- **Provisions for plug-in, optically isolated receivers, drivers, and terminators**
- **Voltate/current levels**
  - **— Input up to 48V**
  - **— Output up to 30V, 60 mA**
- **Common interrupt for up to 8 sources**
- **+5V supply only**

The iSBC 556 Optically Isolated I/O Board provides 48 digital input/output lines with isolation between process application or peripheral device and the system CPU board(s). The iSBC 556 contains two 8255A programmable interface devices, and sockets for user supplied optically isolated drivers, receivers, and input resistor terminators, together with common interrupt logic and inteface circuitry for the system bus. Input signals can be single-ended or differential types with user defined input range (resistor terminator and opto-isolated receiver selection), allowing flexibility in design of voltage and threshold levels. The output allows user selection of Opto-Isolated Darlington Pair which can be used as an output driver either as an open collector or current switch.

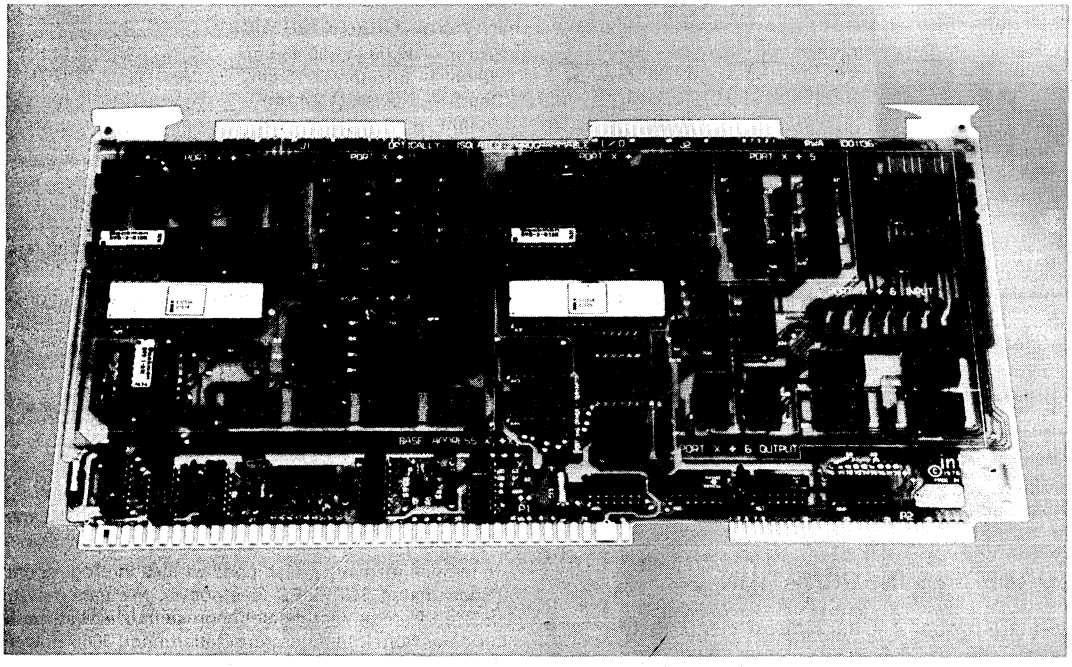


Table 1. I/O Ports Opto-Isolator Receivers, Drivers, and Terminators

Port No. X = I/O Base Address	Type of I/O	Lines (qty)	Resistor Terminator Pac-Rp 16-Pin DIP Bourns 4116R-00 or Equivalent	Dual Opto-Isolator 8-Pin DIP Monsanto MC T66 or Equivalent	Driver 7438 or Equivalent	Pull-Up iSBC® 902
X+0	Input	8	1	4	—	
X+1	Output	8	—		—	
X+2	Input/ Control	8	1		—	
X+4	Input	8	1	4	—	
X+5	Output	8			—	
X+6	Input/ Output Control	8	1 if input		2 if output	2 if input
X+7						

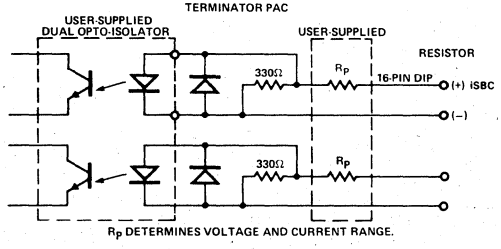
SPECIFICATIONS

Number of Lines

24 input lines
16 output lines
8 programmable lines: 4 input — 4 output

I/O Interface Characteristics

Line-to-Line Isolation — 235V DC or peak AC
Input/Output Isolation — 500V DC or peak AC



Bus Interface Characteristics

All data address and control commands are iSBC 80 bus compatible.

I/O Addressing

Port	8255 #1			Control	8255 #2			Control
	A	B	C		A	B	C	
Address	X+0	X+1	X+2	X+3	X+4	X+5	X+6	X+7

Where:
base address is from 00H to 1FH (jumper selectable)

ORDERING INFORMATION

Part Number	Description
SBC 556	Optically Isolated I/O Board

Connectors

Interface	Pins (qty)	Centers		Mating Connectors
		in.	cm	
P1 iSBC bus	86	0.156		Viking 3KH43/9AMK12
J1 16 fixed input & 8 fixed output lines	50	0.1		3M 3415-000 or TI M312125
J2 8 fixed output, 8 fixed output, & 8 programmable input/output lines	50	0.1		3M 3415-000 or TI M312125

Physical Characteristics

Width — 12.00 in. (30.48 cm)
Height — 6.75 in. (17.15 cm)
Depth — 0.50 in. (1.27 cm)
Weight — 12 oz (397.3 gm)

Electrical Characteristics

Average DC Current

$V_{CC} = +5V \pm 5\%$, 1.0A without user supplied isolated receiver/driver

$I_{CC} = 1.6A$ max with user supplied isolator receiver/driver

Environmental Characteristics

Temperature — 0°C to 55°C
Relative Humidity — 0 to 90%, non-condensing

Reference Manual

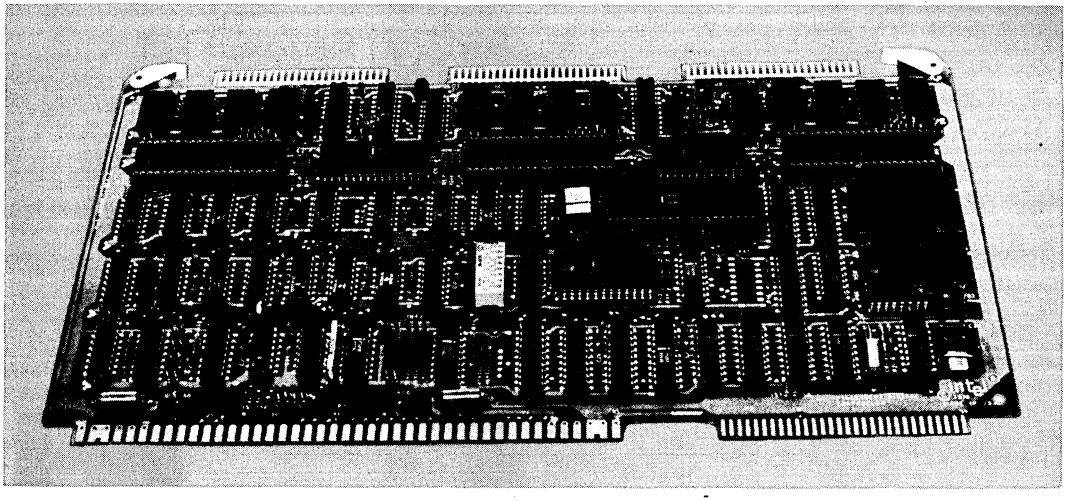
502170 — iSBC 556 Hardware Reference Manual (Order Separately)

Reference manuals are shipped with each product only if designated SUPPLIED (see above). Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

iSBC® 569
INTELLIGENT DIGITAL CONTROLLER

- **Single board digital I/O controller with up to four microprocessors to share the digital input/output signal processing**
- **3 MHz 8085A central control processor**
- **Three sockets for 8041/8741A Universal Peripheral Interface (UPI-41A) for distributed digital I/O processing**
- **Three operational modes**
  - **— Stand-along digital controller**
  - **— MULTIBUS® master**
  - **— Intelligent slave (slave to MULTIBUS® master)**
- **2K bytes of dual port static read/write memory**
- **Sockets for up to 8K bytes of Intel 2758, 2716, 2732 erasable programmable read only memory**
- **48 programmable parallel I/O lines with sockets for interchangeable line drivers or terminators**
- **Three programmable counters**
- **12 levels of programmable interrupt control**
- **Single + 5V supply**
- **MULTIBUS® standard control logic compatible with optional iSBC 80 and iSBC 86 CPU, memory, and I/O expansion boards**

The Intel iSBC 569 Intelligent Digital Controller is a single board computer (8085A based) with sockets for three 8041A/8741A Universal Peripherals Interface chips (UPI-41A). These devices, which are programmed by the user, may be used to offload the 8085A processor from time consuming tasks such as pulse counting, event sensing and parallel or serial digital I/O data formatting with error checking and handshaking. The iSBC 569 board is a complete digital controller with up to four processors on a single 6.75 inches x 12.00 inches (17.15cm x 30.48cm) printed circuit board. The 8085A CPU, system clock, read/write memory, non-volatile memory, priority interrupt logic, programmed timers, MULTIBUS control and interface logic, optional UPI processors and optional line driver and terminators all reside on one board.



FUNCTIONAL DESCRIPTION

Intelligent Digital Controller

Three modes of operation — the iSBC 569 Intelligent Digital Controller is capable of operating in one of three modes; stand alone controller, bus master, or intelligent slave.

Stand alone controller — the iSBC 569 board may function as a stand alone, single board controller with CPU, memory, and I/O elements on a single board. Five volt (+5VDC) only operation allows configuration of low cost controllers with only a single power supply voltage. The on-board 2K bytes RAM and up to 16K bytes ROM/EPROM, as well as the assistance of three UPI-41A processors, allow significant digital I/O control from a single board.

Bus master — in this mode of operation, the iSBC 569 controller may interface with and control iSBC expansion memory and I/O boards, or even other iSBC 569 Intelligent Digital Controllers configured as intelligent slaves (but no additional bus masters).

Intelligent slave — the iSBC 569 controller can perform as an intelligent slave to any 8- or 16-bit MULTIBUS master CPU by offloading the master of digital control related tasks. Preprocessing of data for the master is controlled by the on-board 8085A CPU which coordinates up to three UPI-41A processors. Using the iSBC 569 board as an intelligent slave, multi-channel digital control can be managed entirely on-board, freeing a system master to perform other system functions. The dual port RAM memory allows the iSBC 569 controller to process and store data without MULTIBUS memory contention.

Simplified Programming

By using Intel UPI-41A processors for common tasks such as counting, sensing change of state, printer control and keyboard scanning/debouncing, the user frees up time to work on the more important application programming of machine or process optimization. Controlling the Intel UPI-41A processors becomes a simple task of reading or writing command and data bytes to or from the data bus buffer register on the UPI device.

Central Processing Unit

A powerful Intel 8085A 8-bit CPU, fabricated on a single LSI chip, is the central processor for the iSBC 569 ™ controller. The six general purpose 8-bit registers may be addressed individually or in pairs, providing both single and double precision operations. The program counter can address up to 64K bytes of memory using iSBC expansion boards. The 16-bit stack pointer controls the addressing of an external stack. This stack provides sub-routine nesting bounded only by memory size. The minimum instruction execution time is 1.30 microseconds. The 8085A CPU is software compatible with the Intel 8080A CPU.

Bus Structure

The iSBC 569 Intelligent Digital Controller utilizes a triple bus architecture concept. An internal bus is used for on-board memory and I/O operations. A MULTIBUS interface is available to provide access for all external memory and I/O operations. A dual port bus with controller enables access via the third bus to 2K bytes of static RAM from either the on-board CPU or a system master. Hence, common data may be stored in on-board memory and may be accessed either by the on-board CPU or by system masters. A block diagram of the iSBC 569 functional components is shown in Figure 1.

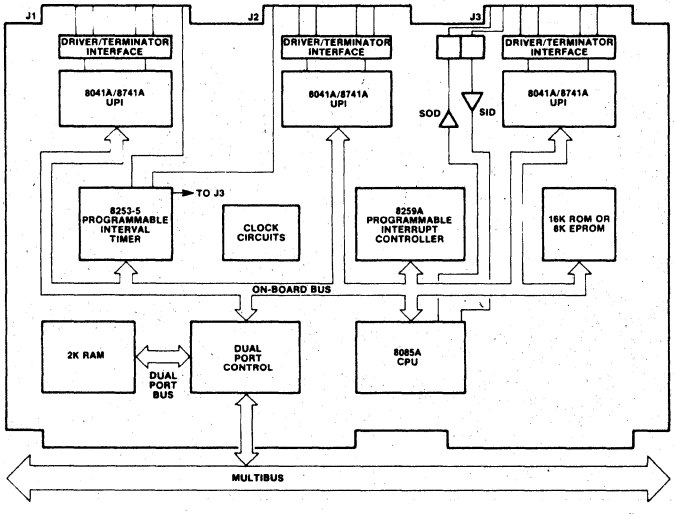


Figure 1. ISBC ®569 Intelligent Digital Controller Block Diagram

RAM Capacity

The iSBC 569 board contains 2K bytes of read/write memory using Intel 2114 static RAMs. RAM accesses may occur from either the iSBC 569 controller or from any other bus master interfaced via the MULTIBUS system bus. The iSBC 569 board provides addressing jumpers to allow the on-board RAM to reside within a one megabyte address space when accessed via the system bus. In addition, a switch is provided which allows the user to reserve a 1K byte segment of on-board RAM for use by the 8085A CPU. This reserved RAM space is not accessible via the system bus and does not occupy any system address space.

EPROM/ROM Capacity

Two sockets for up to 16K bytes of nonvolatile read only memory are provided on the iSBC 569 board. Nonvolatile memory may be added in 1K-byte increments up to a maximum of 2K bytes using Intel 2758 erasable and electrically reprogrammable ROMs (EPROMs); in 2K-byte increments up to a maximum of 4K bytes using Intel 2316 ROMs or 2716 EPROMs; in 4K byte increments up to 8K bytes maximum using Intel 2732 EPROMs; or in 8K-byte increments up to 16K bytes maximum using Intel 2364 ROMs (both sockets must contain same type ROM/EPROM). All on-board ROM/EPROM operations are performed at maximum processor speed.

Universal Peripheral Interfaces (UPI-41A)

The iSBC 569 Intelligent Digital Controller board provides three sockets for user supplied Intel 8041A/8741A Universal Peripheral Interface (UPI-41A) chips. Sockets are also provided for the associated line drivers and terminators for the UPI I/O ports. The UPI-41A processor is a single chip microcomputer containing a CPU, 1K bytes of ROM (8041A) or EPROM (8741A), 64 bytes of RAM, 16 programmable I/O lines, and an 8-bit timer/event counter. Special interface registers included in the chip allow the UPI-41A processor to function as a slave processor to the iSBC 569 controller board's 8085A CPU. The UPI processor allows the user to specify algorithms for controlling peripherals directly thereby freeing the 8085A for other system functions. For additional information, including UPI-41A instructions, refer to the UPI-41 User's Manual (Manual No. 9800504).

Programmable Timers

The iSBC 569 Intelligent Digital Controller board provides three independently programmable interval timer/counters utilizing one Intel 8253 Programmable Interval Timer (PIT). The Intel 8253 PIT provides three 16-bit BCD or binary interval timer/counters. Each timer may be used to provide a time reference for each UPI™ processor or for a group of UPI processors. The output of each timer also connects to the 8259A Programmable Interrupt Controller (PIC) providing the capability of timed interrupts. All gate inputs, clock inputs, and timer outputs of the 8253 PIT are available at the I/O ports for external access.

Timer Functions — In utilizing the iSBC 569 controller, the systems designer simply configures, via software, each timer to meet systems requirements. The 8253 PIT modes are listed in Table 1. The contents of each counter may be read at any time during system operation with simple read operations for event counting applications. The contents of each counter can be read "on-the-fly" for time stamping events or time clock referenced program initiations.

Table 1. 8253 Programmable Timer Functions

Function	Operation
Interrupt on terminal count	When terminal count is reached, an interrupt request is generated.
Programmable one-shot	Output goes low upon receipt of an external trigger edge or software command and returns high when terminal count is reached. This function is retriggerable.
Rate generator	Divide by N counter. The output will go low for one input clock cycle, and the period from one low-going pulse to the next is N times the input clock period.
Square-wave rate generator	Output will remain high until one-half the count has been completed, and go low for the other half of the count.
Software triggered strobe	Output remains high until software loads count (N). N counts after count is loaded, output goes low for one input clock period.
Hardware triggered strobe	Output goes low for one clock period N counts after rising edge on counter trigger input. The counter is retriggerable.
Event counter	On a jumper selectable basis, the clock input becomes an input from the external system. CPU may read the number of events occurring after the counting "window" has been enabled or an interrupt may be generated after N counts occur in the system.

Interrupt Capability

The iSBC 569 Intelligent Digital Controller provides interrupt service for up to 12 interrupt sources. Any of the 12 sources may interrupt the on-board processor. Four interrupt levels are handled directly by the 8085A CPU and eight levels are serviced from an Intel 8259A Programmable Interrupt Controller (PIC) routing an interrupt request output to the INTR input of the 8085A.

8085A Interrupt — Each of four direct 8085A interrupt inputs has a unique vector memory address. An 8085A jump instruction at each of these addresses then provides software linkage to interrupt service routines located independently anywhere in the memory.

8259A Interrupts — The eight interrupt sources originate from both on-board controller functions and the system bus:

UPI-41A Processors — one interrupt from each of three UPI processor sockets.

8253 PIT — one interrupt from each of three timer outputs.

MULTIBUS System Bus — one of eight MULTIBUS interrupt lines may be jumpered to either of two 8259A PIC interrupt inputs.

Programmable Reset — The iSBC 569 Intelligent Digital Controller board has a programmable output latch used to control on-board functions. Three of the outputs are connected to separate UPI-41A RESET inputs. Thus, the user can reset any or all of the UPI-41A processors under software control. A fourth latch output may be used to generate an interrupt request onto the MULTIBUS interrupt lines. A fifth latch output is connected to a light-emitting diode which may be used for diagnostic purposes.

Expansion Capabilities

When the iSBC 569 controller is used as a single board digital controller, memory and I/O capacity may be expanded using Intel MULTIBUS compatible expansion boards. In this mode, no other bus masters may be in the system. Memory may be expanded to a 64K byte capacity by adding user specified combinations of RAM boards, EPROM boards, or combination boards. Input/output capacity may be increased by adding I/O expansion boards. Multiple iSBC 569 boards may be included in an expanded system using one iSBC 569 Intelligent Digital Controller as the system master and additional controllers as intelligent slaves.

Intelligent Slave Programming

When used as an intelligent slave, the iSBC 569 controller appears as an additional RAM memory module. System bus masters communicate with the iSBC 569 board as if it were just an extension of system memory. To simplify this communication, the user has been given some specific tools:

Flag Interrupt — The Flag Interrupt is generated any time a write command is performed by an off-board CPU to the first location of iSBC 569 RAM. This interrupt provides a means for the master CPU to notify the iSBC 569 controller that it wished to establish a communications sequence. The flag interrupt is cleared when the on-board processor reads the first location of its RAM. In systems with more than one intelligent slave, the flag interrupt provides a unique interrupt to each slave outside the normal MULTIBUS interrupt lines (INT0/-INT7/).

RAM — The on-board 2K byte RAM area that is accessible to both an off-board CPU and the on-board 8085A may be configured for system access on any 2K boundary.

MULTIBUS Interrupts — The third tool to improve system operation as an intelligent slave is access to the MULTIBUS interrupt lines. The iSBC 569 controller can both respond to interrupt signals from an off-board CPU, and generate an interrupt to the off-board CPU via the system bus.

System Development Capability

Software development for the iSBC 569 Intelligent Digital Controller board is supported by the Intellec® Microcomputer Development System including a resident macroassembler, text editor, system monitor, a linker, object code locator, and Library Manager. In addition, both PL/M and FORTRAN language programs can be compiled to run on the iSBC 569 board. A unique in-circuit emulator (ICE-85™) option provides the capability of developing and debugging software directly on the iSBC 569 board. This greatly simplifies the design, development, and debug of iSBC 569 system software.

SPECIFICATIONS

8085A CPU

Word Size — 8, 16 or 24 bits

Cycle Time — 1.30 μsec ± .1% for fastest executable instruction; i.e., four clock cycles.

Clock Rate — 3.07 MHz ± .1%

System Access Time

Dual port memory — 725 nsec

Memory Capacity

On-board ROM/EPROM — 2K, 4K, 8K, or 16K bytes of user installed ROM or EPROM

On-board RAM — 2K bytes of static RAM. Fully accessible from on-board 8085A. Separately addressable from system bus.

Off-board expansion — up to 64K bytes of EPROM/ROM or RAM capacity.

I/O Capacity

Parallel-Timers — Three timers, with independent gate input, clock input, and timer output user-accessible. Clock inputs can be strapped to an external source or to an on-board 1.3824 MHz reference. Each timer is connected to a 8259A Programmable Interrupt Controller and may also be optionally connected to UPI processors.

UPI-I/O — Three UPI-41A interfaces, each with two 8-bit I/O ports plus the two UPI Test Inputs. The 8-bit ports are user-configurable (as inputs or outputs) in groups of four.

Serial — 1 TTL compatible serial channel utilizing SID and SOD lines of on-board 8085A CPU

On-Board Addressing

All communications to the UPI-41A processors, to the programmable reset latch, to the timers, and to the interrupt controller are via read and write commands from the on-board 8085A CPU.

Memory Addressing

On-board ROM/EPROM — 0-07FF (using 2758 EPROMs); 0-0FFF (using 2716 EPROMs or 2316 ROMs); 0-1FFF (using 2732 EPROMs); 0-3FFF (using the 2364 ROMs)

On-board RAM — 8000-87FF System access — any 2K increment 00000-FF800 (switch selectable); 1K bytes may be disabled from bus access by switch selection.

I/O Addressing

Source	Addresses
8253	0E0H-0E3H
UPI0	0E4H-0E5H
UPI1	0E6H-0E7H
UPI2	0E8H-0E9H
PROGRAMMABLE RESET	0EAH-0EBH
8259A	0ECH-0EDH

Timer Specifications

Input frequencies — jumper selectable reference

Internal: 1.3824 MHz ± .1% (.723 μsec, nominal)

External: User supplied (2 MHz maximum)

Output Frequencies (at 1.3824 MHz)

Function	Min[1]	Max[1]
Real-time interrupt interval	1.45 μsec	47.4 msec
Rate Generator (frequency)	21.09 Hz	691.2 KHz

1. Single 16-bit binary count

Interfaces

MULTIBUS™ Interface — All signals compatible with iSBC and MULTIBUS architecture

Parallel I/O — All signals TTL compatible

Interrupt Requests — All TTL compatible

Timer — All signals TTL compatible

Serial I/O — All signals TTL compatible

Connectors

Interface	Pins (qty)	Centers (in.)	Mating Connectors
Bus	86	0.156	Viking 3KH43/9AMK12
Parallel I/O	50	0.1	3M 3415-000 or TI H312125

Physical Characteristics

Width — 30.48 cm (12.00 inches)

Depth — 17.15 cm (6.75 inches)

Thickness — 1.27 cm (0.50 inch)

Weight — 3.97 gm (14 ounces)

Electrical Characteristics

DC Power Requirements — +5V @ 2.58A with no optional devices installed. For each 8741A add 135 mA. For each 220/330 resistor network, add 60 mA. Add the following for each EPROM/ROM installed.

Type	+5.0V Current Requirement	
	1ROM	2ROMS
2758	100 mA	125 mA
2716	100 mA	125 mA
2316E	120 mA	240 mA
2732	40 mA	55 mA
2364	40 mA	55 mA

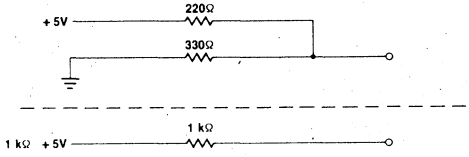
Line Drivers and Terminators

I/O Drivers — The following line drivers are all compatible with the I/O driver sockets on the iSBC 569 Intelligent Digital Controller.

Driver	Characteristic	Sink Current (mA)
7438	I,OC	48
7437	I	48
7432	NI	16
7426	I,OC	16
7409	NI,OC	16
7408	NI	16
7403	I,OC	16
7400	I	16

Note I = inverting; NI = non-inverting; OC = open collector.

I/O Terminators — 220Ω/330Ω divider or 1 kΩ pullup (DIP) - user supplied



Environmental Characteristics

Operating Temperature — 0°C to 55°C (32°F to 131°F)

Relative Humidity — To 90% without condensation

Reference Manual

502180 — iSBC 569 Intelligent Digital Controller Board Hardware Reference Manual (NOT SUPPLIED)

Reference manuals are shipped with each product only if designated SUPPLIED (see above). Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

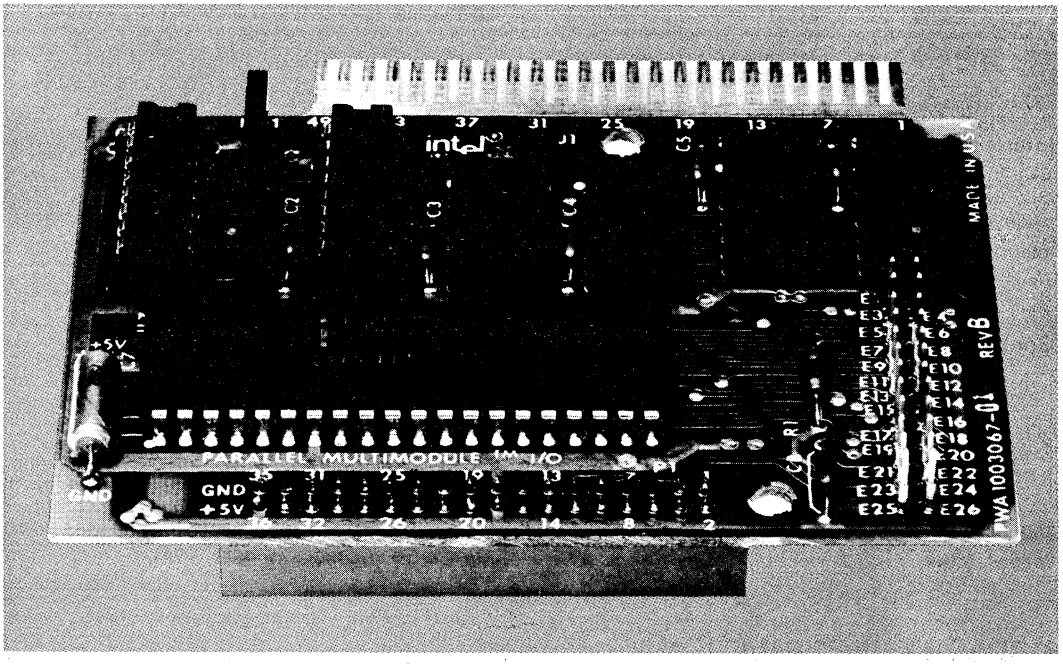
ORDERING INFORMATION

Part Number	Description
SBC 569	Intelligent Digital Controller

iSBX™ 350
PARALLEL I/O MULTIMODULE™BOARD

- **iSBX™ bus compatible I/O expansion**
- **24 programmable I/O lines with sockets for interchangeable line drivers and terminators**
- **Three jumper selectable interrupt request sources**
- **Accessed as I/O port locations**
- **Single +5V low power requirement**
- **iSBX™ bus on-board expansion eliminates MULTIBUS® system bus latency and increases system throughput**

The Intel® iSBX 350 Parallel I/O MULTIMODULE Board is a member of Intel's line of iSBX bus compatible MULTIMODULE products. The iSBX MULTIMODULE board plugs directly into any iSBX bus compatible host board offering incremental on-board expansion. The iSBX 350 module provides 24 programmable I/O lines with sockets for interchangeable line drivers and terminators. The iSBX board is closely coupled to the host board through the iSBX bus, and as such, offers maximum on-board performance and frees MULTIBUS system traffic for other system resources. In addition, incremental power dissipation is minimal requiring only 1.6 watts (not including optional driver/terminators).



AFN-01382A

FUNCTIONAL DESCRIPTION

Programmable Interface

The iSBX 350 module uses an Intel® 8255A-5 Programmable Peripheral Interface (PPI) providing 24 parallel I/O lines. The base-board system software is used to configure the I/O lines in any combination of unidirectional input/output and bidirectional ports indicated in Table 1. Therefore, the I/O interface may be customized to meet specific peripheral requirements. In order to take full advantage of the large number of possible I/O configurations, sockets are provided for interchangeable I/O line drivers and terminators. Hence, the flexibility of the I/O interface is further enhanced by the capability of selecting the appropriate combination of optional line drivers and terminators to provide the required sink current, polarity, and driver/termination characteristics for each application. In addition, inverting bidirectional bus drivers (8226) are provided on sockets to allow convenient optional replacement to non-inverting drivers (8216). The 24 programmable I/O lines, signal ground, and +5 volt power (jumper configurable) are brought to a 50-pin edge connector that mates with flat, woven, or round cable.

Interrupt Request Generation

Interrupt requests may originate from three jumper selectable sources. Two interrupt requests can be automatically generated by the PPI when a byte of information is ready to be transferred to the base board CPU (i.e., input buffer is full) or a byte of information has been transferred to a peripheral device (i.e., output buffer is empty). A third interrupt source may originate directly from the user I/O interface (J1 connector).

Installation

The iSBX 350 module plugs directly into the female iSBX connector on the host board. The module is then secured at one additional point with nylon hardware to insure the mechanical security of the assembly (see Figure 1 and Figure 2).

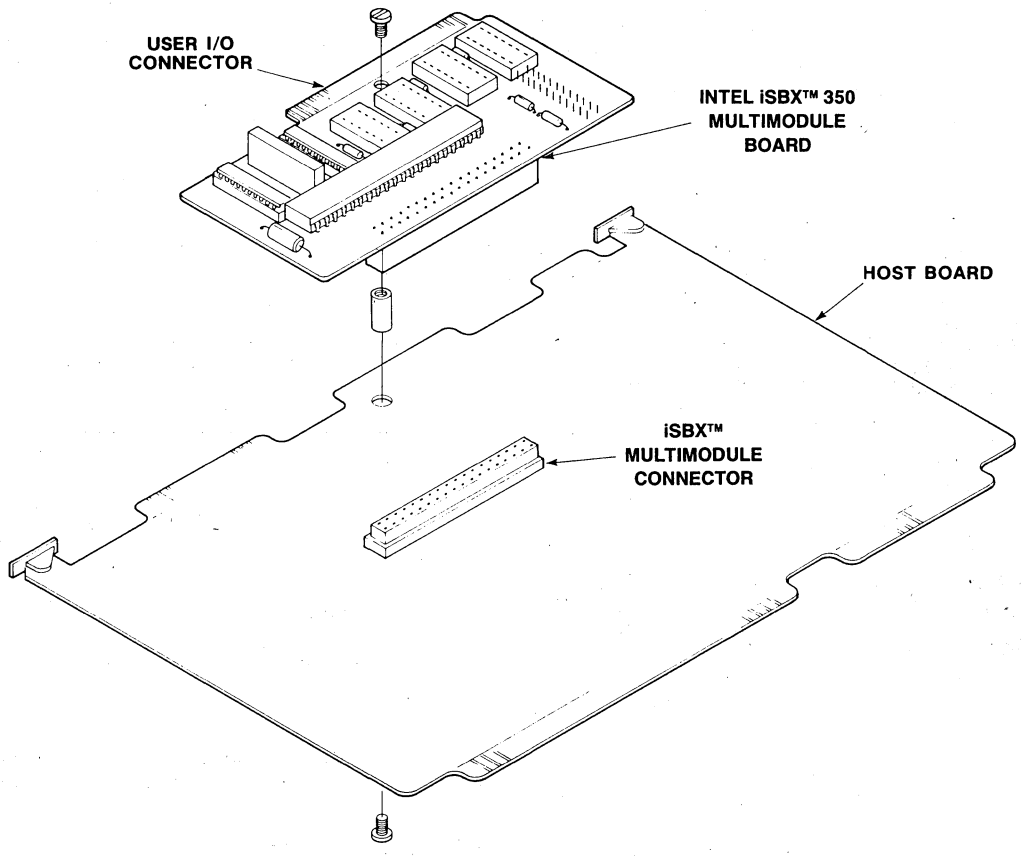


Figure 1. Installation of iSBX™ 350 Module on a Host Board

AFN-01382A

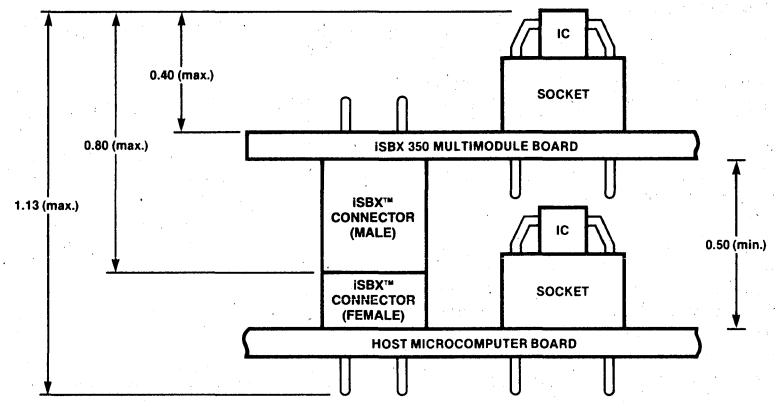


Figure 2. Mounting Clearances (inches)

Table 1. Input/Output Port Modes of Operation

Port	Lines (qty)	Mode of Operation					Control
		Unidirectional				Bidirectional	
		Input		Output			
		Unlatched	Latched & Strobed	Latched	Latched & Strobed		
A	8	X	X	X	X	X	
B	8	X	X	X	X		
C	4	X		X			X[1]
	4	X		X			X[1]

NOTE:

1. Part of port C must be used as a control port when either port A or port B are used as a latched and strobed input or a latched and strobed output port or port A is used as a bidirectional port.

SPECIFICATIONS

Word Size

Data — 8 Bits

I/O Addressing

8255A-5 Ports	iSBX 350 Address
Port A	X0 or X4
Port B	X1 or X5
Port C	X2 or X6
Control	X3 or X7
Reserved	X8 to XF

NOTE:

The first digit of each port I/O address is listed as "X" since it will change dependent on the type of host iSBC microcomputer used. Refer to the Hardware Reference Manual for your host iSBC microcomputer to determine the first digit of the port address.

I/O Capacity

24 programmable lines (see Table 1)

Access Time

Read — 250 ns max.
Write — 300 ns max.

NOTE:

Actual transfer speed is dependent upon the cycle time of the host microcomputer.

Interrupts

Interrupt requests may originate from the programmable peripheral interface (2) or the user specified I/O (1).

Interfaces

iSBX™ Bus — All signals TTL compatible
Parallel I/O — All signals TTL compatible

Parallel Interface Connectors

Interface	No. of Pairs/ Pins	Centers (in.)	Connector Type	Vendor	Vendor Part No.
Parallel I/O Connector	25/50	0.1	Female	3M	3415-0001 with Ears
Parallel I/O Connector	25/50	0.1	Female, Soldered	GTE Sylvania	6AD01251A1DD

Note: Connector compatible with those listed may also be used.

Line Drivers and Terminators

I/O Drivers — The following line drivers and terminators are all compatible with the I/O driver sockets on the iSBX 350.

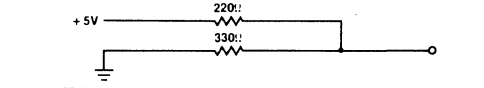
Driver	Characteristic	Sink Current (mA)
7438	I, OC	48
7437	I	48
7432	NI	16
7426	I, OC	16
7409	NI, OC	16
7408	NI	16
7403	I, OC	16
7400	I	16

Note:
I = Inverting, NI = Non-Inverting, OC = Open Collector

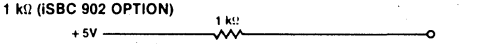
Port 1 has 25 mA totem pole drivers and 1 kΩ terminators.

I/O Terminators — 220Ω/330Ω divider or 1 kΩ pull up.

220Ω/330Ω (iSBC 901 OPTION)



1 kΩ (iSBC 902 OPTION)



Physical Characteristics

Width — 7.24 cm (2.85 in.)
Length — 9.40 cm (3.70 in.)
Height* — 2.04 cm (0.80 in.) iSBX 350 Board
— 2.86 cm (1.13 in.) iSBX 350 Board + Host Board
Weight — 51 gm (1.79 oz)

*See Figure 2.

Electrical Characteristics

DC Power Requirements

Power Requirement	Configuration
+5V @ 320 mA	Sockets XU3, XU4, XU5, and XU6 empty (as shipped).
+5V @ 500 mA	Sockets XU3, XU4, XU5, and XU6 contain 7438 buffers.
+5V @ 620 mA	Sockets XU3, XU4, XU5, and XU6 contain iSBC 901 termination devices.

Environmental

Operating Temperature — 0°C to 55°C

Reference Manual

9803191-01 — iSBX 350 Parallel I/O MULTIMODULE Manual (NOT SUPPLIED)

Reference Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Ave., Santa Clara, California 95051.

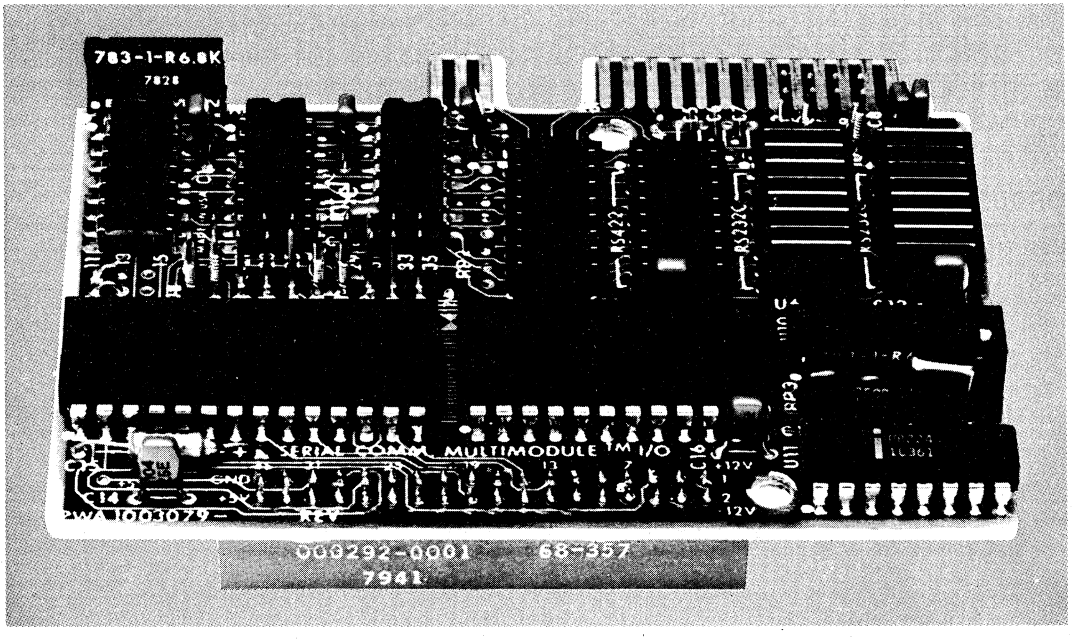
ORDERING INFORMATION

Part Number	Description
SBX 350	Parallel I/O MULTIMODULE Board

iSBX™351
SERIAL I/O MULTIMODULE™ BOARD

- **iSBX ™ bus compatible I/O expansion**
- **Programmable synchronous/asynchronous communications channel with RS232C or RS449/422 interface**
- **Software programmable baud rate generator**
- **Two programmable 16-bit BCD or binary timers/event counters**
- **Four jumper selectable interrupt request sources**
- **Accessed as I/O port locations**
- **Low power requirements**
- **Single +5V when configured for RS449/422 interface**
- **iSBX bus on-board expansion eliminates MULTIBUS ® system bus latency and increases system throughput**

The Intel® iSBX 351 Serial I/O MULTIMODULE board is a member of Intel's new line of iSBX bus compatible MULTIMODULE products. The iSBX MULTIMODULE board plugs directly into any iSBX bus compatible host board offering incremental on-board I/O expansion. The iSBX 351 module provides one RS232C or RS449/422 programmable synchronous/asynchronous communications channel with software selectable baud rates. Two general purpose programmable 16-bit BCD or binary timers/event counters are available to the host board to generate accurate time intervals under software control. The iSBX board is closely coupled to the host board through the iSBX bus, and as such, offers maximum on-board performance and frees MULTIBUS system traffic for other system resources. In addition, incremental power dissipation is minimal requiring only 3.0 watts (assumes RS232C interface).



FUNCTIONAL DESCRIPTION

Communications Interface

The iSBX 351 module uses the Intel® 8251A Universal Synchronous/Asynchronous Receiver/Transmitter (USART) providing one programmable communications channel. The USART can be programmed by the system software to individually select the desired asynchronous or synchronous serial data transmission technique (including IBM Bi-Sync). The mode of operation (i.e. synchronous or asynchronous), data format, control character format, parity, and baud rate are all under program control. The 8251A provides full duplex, double buffered transmit and receive capability. Parity, overrun, and framing error detection are all incorporated in the USART. The command lines, serial data lines, and signal ground lines are brought out to a double edge connector configurable for either an RS232C or RS449/422 interface (see Figure 3). In addition, the iSBX 351 module is jumper configurable for either point-to-point or multidrop network connection.

16-Bit Interval Timers

The iSBX 351 module uses an Intel 8253 Programmable Interval Timer (PIT) providing 3 fully programmable and independent BCD and binary 16-bit

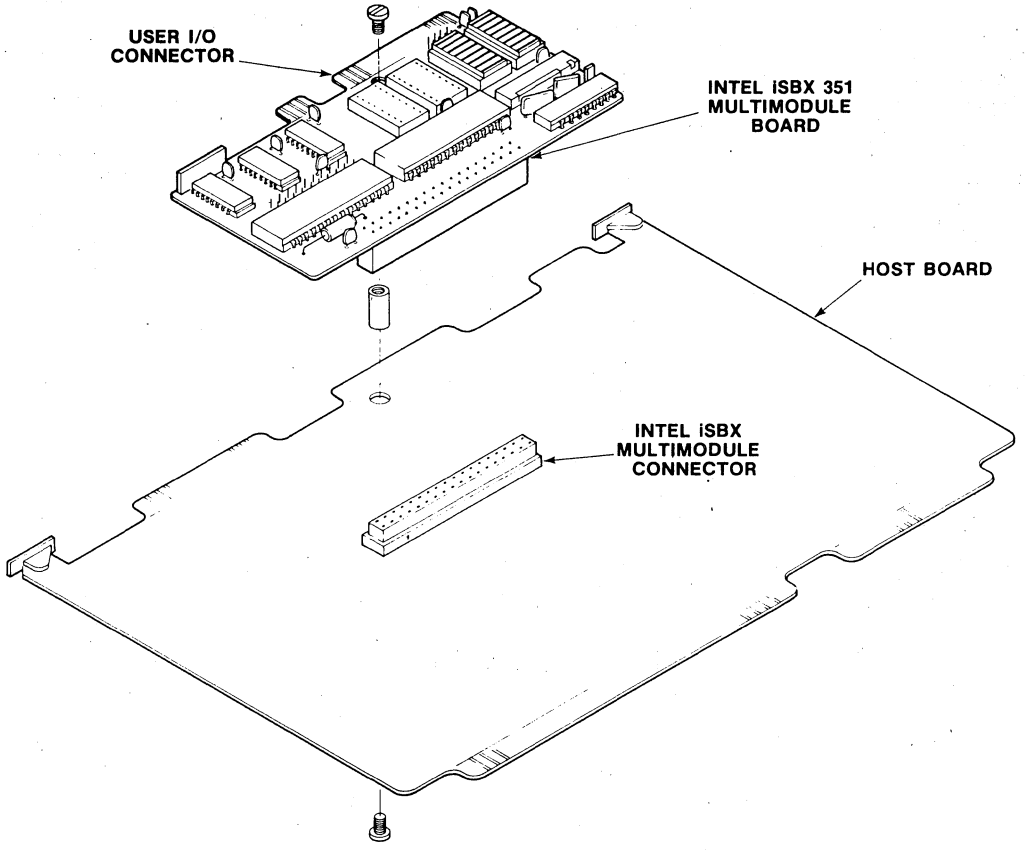


Figure 1. Installation of iSBC® 351 Module on a Host Board

interval timers. One timer is available to the system designer to generate baud rates for the USART under software control. Routing for the outputs from the other two counters is jumper selectable to the host board. In utilizing the iSBX 351 module, the systems designer simply configures, via software, each timer independently to meet system requirements. Whenever a given baud rate or time delay is needed, software commands the programmable timers to select the desired function. The functions of the timers are shown in Table 1. The contents of each counter may be read at any time during system operation.

Interrupt Request Lines

Interrupt requests may originate from four sources. Two interrupt requests can be automatically generated by the USART when a character is ready to be transferred to the host board (i.e. receive buffer is full) or a character has been transmitted (i.e. transmit buffer is empty). In addition, two jumper selectable requests can be generated by the programmable timers.

Installation

The iSBX 351 module plugs directly into the female iSBX connector on the host board. The module is then secured at one additional point with nylon hardware to insure the mechanical security of the assembly (see Figures 1 and 2).

Table 1. Programmable Timer Functions

Function	Operation
Interrupt on terminal count	When terminal count is reached, an interrupt request is generated. This function is useful for generation of real-time clocks.
Programmable one-shot	Output goes low upon receipt of an external trigger edge and returns high when terminal count is reached. This function is retriggerable.
Rate generator	Divide by N counter. The output will go low for one input clock cycle, and the period from one low going pulse to the next is N times the input clock period.
Square-wave rate generator	Output will remain high until one-half the count has been completed, and go low for the other half of the count.
Software triggered strobe	Output remains high until software loads count (N). N counts after count is loaded, output goes low for one input clock period.
Hardware triggered strobe	Output goes low for one clock period N counts after rising edge counter trigger input. The counter is retriggerable.
Event counter	On a jumper selectable basis, the clock input becomes an input from the external system. CPU may read the number of events occurring after the counting "window" has been enabled or an interrupt may be generated after N events occur in the system.

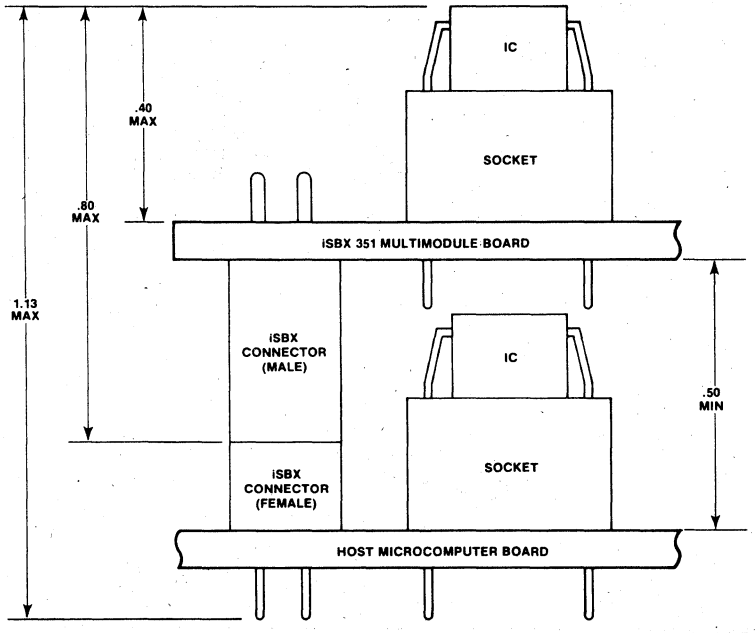


Figure 2. Mounting Clearances (Inches)

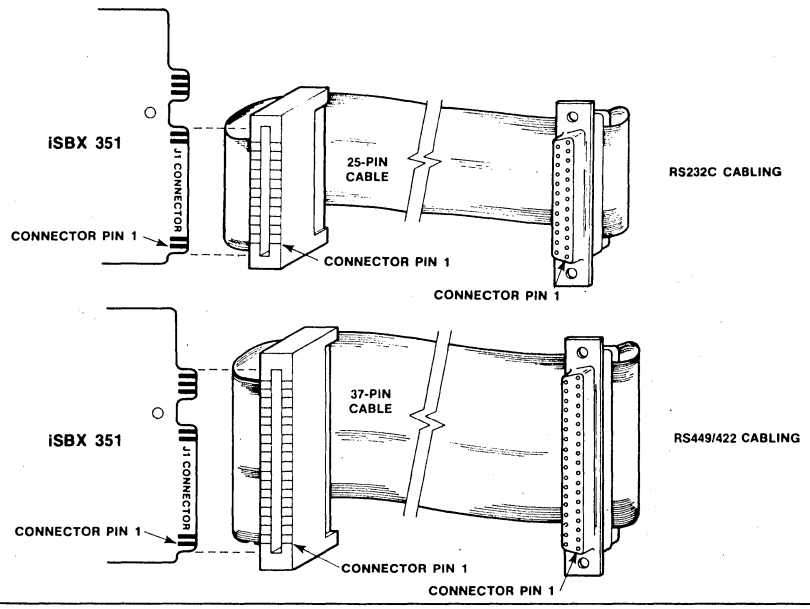


Figure 3. Cable Construction and Installation for RS232C and RS449/422 Interface

SPECIFICATIONS

Word Size

Data — 8 bits

I/O Addressing

I/O Address for an 8-bit host	I/O Address for a 16-bit host	Chip Select	Function
X0, X2, X4 or X6	Y0, Y4, Y8 or YC	8251A USART	Write: Data Read: Data
X1, X3, X5 or X7	Y2, Y6, YA or YE	MCS0/Activated (True)	Write: Mode or command Read: Status
X8 or XC	Z0 or Z8	8253 PIT	Write: Counter 0 Load: Count (N) Read: Counter 0
X9 or XD	Z2 or ZA	MCS1/Activated (True)	Write: Counter 1 Load: Count N Read: Counter 1
XA or XE	Z4 or ZC		Write: Counter 2 Load: Count (N) Read: Counter 2
XB or XF	Z6 or ZE		Write: Control Read: None

Notes: X = The iSBX base address that activates MSC0/ & MSC1 for an 8-bit Host.

Y = The iSBX base address that activates MCS0/ for a 16-bit host.

Z = The iSBX base address that activates MCS1/ for a 16-bit host.

The first digit, X, Y, or Z, is always a variable, since it will depend on the type of host micorcomputer used. Refer to the Hardware Reference Manual for your host microcomputer to determine the first digit of the I/O base address.

NOTE: The first digit of each port I/O address is listed as "X" since it will change depending on the type of host iSBC microcomputer used. Refer to the Hardware Reference Manual for your host iSBC microcomputer to determine the first digit of the I/O address.

Access Time

Read — 250 nsec max

Write — 300 nsec max

Note Actual transfer speed is dependent upon the cycle time of the host microcomputer.

Serial Communications

Synchronous — 5 - 8-bit characters; internal character synchronization; automatic sync insertion; even, odd or no parity generation/detection.

Asynchronous — 5 - 8-bit characters; break character generation and detection; 1, 1½, or 2 stop bits; false start bit detection; even, odd or no parity generation/detection.

Sample Baud Rate:

8253 PIT Frequency[1] (kHz, Software Selectable)	8251 USART Baud Rate (Hz)[2]	
	Synchronous	Asynchronous
		÷ 16 ÷ 64
307.2	—	19200 4800
153.6	—	9600 2400
76.8	—	4800 1200
38.4	38400	2400 600
19.2	19200	1200 300
9.6	9600	600 150
4.8	4800	300 75
2.4	2400	150 —
1.76	1760	110 —

NOTES: 1. Frequency selected by I/O writes of appropriate 16-bit frequency factor to Baud Rate Register.

2. Baud rates shown here are only a sample subset of possible software-programmable rates available. Any frequency from 18.75 Hz to 614.4 kHz may be generated utilizing on-board crystal oscillator and 16-bit Programmable Interval Timer (used here as frequency divider).

Interval Timer and Baud Rate Generator

Input Frequency (selectable):

1.23 MHz ±0.1% (.813 μsec period nominal)

153.6 kHz ±0.1% (6.5 μsec period nominal)

Output Frequency:

	Rate Generator (Frequency)		Real-Time Interrupt (Interval)	
	Min.	Max.	Min.	Max.
Single Timer[1]	18.75 Hz	614.4 kHz	1.63 μsec	53.3 msec
Single Timer[2]	2.34 Hz	76.8 kHz	13.0 μsec	426.7 msec
Dual Timer[3] (Counters 0 and 1 in series)	0.000286 Hz	307.2 kHz	3.26 μsec	58.25 min
Dual Timer[4] (Counters 0 and 1 in series)	0.0000358 Hz	38.4 kHz	26.0 μsec	7.77 hrs

NOTES: 1. Assuming 1.23 mHz clock input.
2. Assuming 153.6 kHz clock input.
3. Assuming Counter 0 has 1.23 mHz clock input.
4. Assuming Counter 0 has 153.6 kHz clock input.

Interrupts

Interrupt requests may originate from the USART (2) or the programmable timer (2).

Interfaces

iSBX Bus — all signals TTL compatible.

Serial — configurable for EIA Standards RS232C or RS449/422

EIA Standard RS232C signals provided and supported:
- Clear to Send (CTS)
- Data Set Ready (DSR)
- Data Terminal Ready (DTR)
- Request to Send (RTS)
- Receive Clock (RXC)
- Receive Data (RXD)
- Transmit Clock (DTE TXC)
- Transmit Data (TXD)

EIA Standard RS449/422 signals provided and supported:
- Clear to Send (CS)
- Data Mode (DM)
- Terminal Ready (TR)
- Request to Send (RS)
- Receive Timing (RT)
- Receive Data (RD)
- Terminal Timing (TT)
- Send Data (SD)

Physical Characteristics

Width — 7.24 cm (2.85 inches)
Length — 9.40 cm (3.70 inches)
Height* — 2.04 cm (0.80 inches) iSBX 351 Board
— 2.86 cm (1.13 inches) iSBX 351 Board and Host Board
Weight — 51 grams (1.79 ounces)
* (See Figure 2)

Serial Interface Connectors

Configuration	Mode[2]	MULTIMODULE Edge Connector	Cable	Connector[8]
RS232C	DTE	26-pin[5], 3M-3462-0001	3M[3]-3349/25	25-pin[7], 3M-3482-1000
RS232C	DCE	26-pin[5], 3M-3462-0001	3M[3]-3349/25	25-pin[7], 3M-3483-1000
RS449	DTE	40-pin[6], 3M-3464-0001	3M[4]-3349/37	37-pin[1], 3M-3502-1000
RS449	DCE	40-pin[6], 3M-3464-0001	3M[4]-3349/37	37-pin[1], 3M-3503-1000

NOTES: 1. Cable housing 3M-3485-4000 may be used with the connector.
2. DTE — Data Terminal mode (male connector), DCE — Data Set mode (female connector).
3. Cable is tapered at one end to fit the 3M-3462 connector.
4. Cable is tapered to fit 3M-3464 connector.
5. Pin 26 of the edge connector is not connected to the flat cable.
6. Pins 37, 39, and 40 of the edge connector are not connected to the flat cable.
7. May be used with cable housing 3M-3485-1000.
8. Connectors compatible with those listed may also be used.

Electrical Characteristics

DC Power Requirements

Mode	Voltage	Amps (Max.)
RS232C	+5V ±0.25V	460 mA
	+12V ±0.6V	30 mA
	−12V ±0.6V	30 mA
RS449/422	+5V ±0.25V	530 mA

Environmental Characteristics

Temperature — 0 - 55°C, free moving air across the base board and MULTIMODULE board.

Reference Manual

9803190-01 — iSBX 351 Serial I/O MULTIMODULE Manual (NOT SUPPLIED)

Reference Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Ave., Santa Clara, California, 95051.

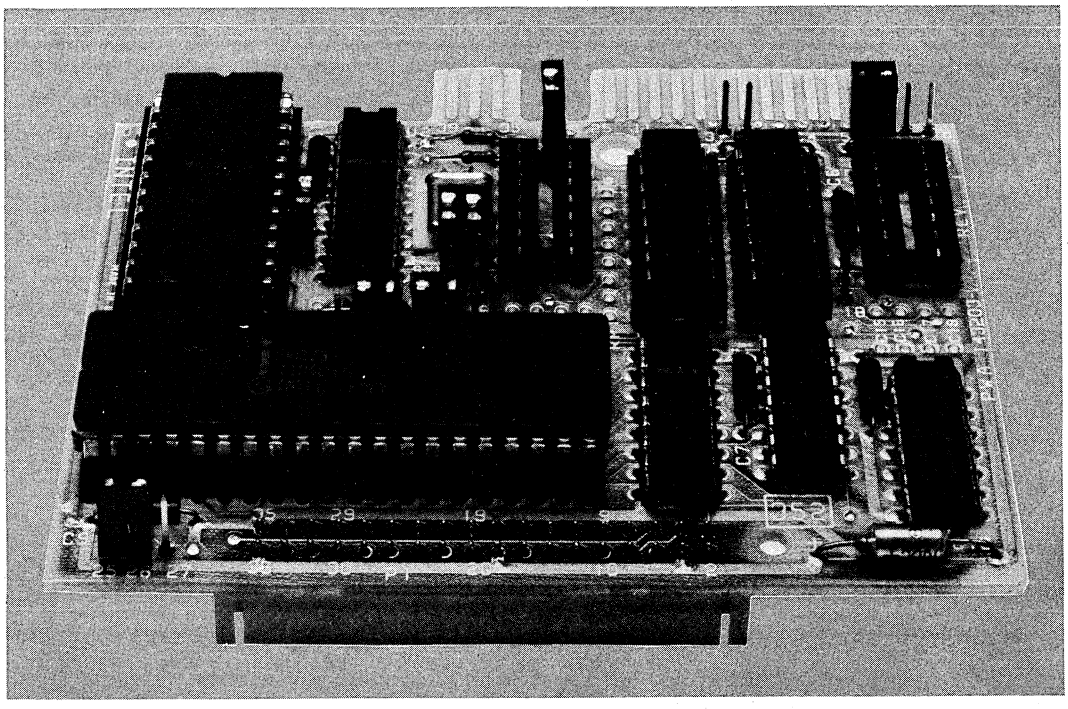
ORDERING INFORMATION

Part Number	Description
SBX 351	Serial I/O MULTIMODULE Board

iSBX™ 352
BIT SERIAL COMMUNICATIONS MULTIMODULE™ BOARD

- **Provides an HDLC/SDLC half/full-duplex communications channel for iSBX™ bus compatible micro-computers**
- **Supports RS232C (including modem support) or RS449/422A interface**
- **Single +5V when configured for RS449/422A interface**
- **Software programmable baud rate generation up to 64K baud synchronous and 9.6K baud self-clocking**
- **Supports synchronous or self-clocking NRZI point-to-point, multidrop and self-clocking NRZI SDLC loop data link interfaces**

The Intel iSBX 352 Bit Serial Communications MULTIMODULE board offers incremental on-board I/O expansion support for ISO/CCITT's HDLC or IBM's SDLC communication. Plugging directly into any iSBX bus compatible host board, the iSBX 352 module provides one RS232C or RS449/422A programmable bit serial communications channel with software selectable baud rates (up to 64K baud for half-duplex synchronous operations). Data link interfaces supported are: synchronous point-to-point, multidrop and SDLC loop. The phase lock loop feature provides NRZI self-clocking 9.6K baud operation.



October, 1981
Order Number: 210218-001

FUNCTIONAL DESCRIPTION

Communications Interface

The iSBX 352 module uses the Intel 8273 Programmable HDLC/SDLC Protocol Controller. The iSBX 352 module provides one bit-serial communications channel for iSBX bus compatible host microcomputers. (See Figure 1.) An iSBC microcomputer or MULTIBUS-based application is easily connected to an HDLC/SDLC point-to-point, multidrop, or an SDLC loop configuration.

The High-Level Data Link Control (HDLC) is the International Standards Organization (ISO) standard discipline used to implement X.25 packet switching communications. The Synchronous Data Link Control (SDLC) is an IBM communication protocol used to implement the System Network Architecture (SNA). Both protocols, HDLC and SDLC, are bit oriented, code independent, and support full-duplex operations.

Data Link Interface

The control lines, serial data lines and signal ground lines are brought out to the double edge connector of the iSBX 352 module and are configurable for RS232C or RS449/422A interface (see Figure 2).

Addressing an iSBX 352 board by using a port address, the program performs the 8-bit data transfer required, using buffered or non-buffered transmit/receive and abort sequences.

Serial data transfer control is provided by the 8273 controller of the iSBX 352 module which interfaces the parallel iSBX bus to the serial channel. During a transmit sequence, the iSBX 352 module accepts data and commands from the iSBX bus interface, translates and formats the data into HDLC/SDLC protocol formats, provides the proper RS232C or RS422A interface control signals, and passes data onto the serial channel. The receive operation is the inverse of the previous sequence.

Data Link Configurations

The supported data link configurations are shown in Table 1. The following example configurations provide an overview and a figure for five typical data link configurations:

Table 1. iSBX™ 352 Supported Configurations

Connection	Synchronous Modem	Synchronous Direct	Asynchronous Modem*	Asynchronous Direct
point-to-point	X	X	X	X
multidrop	X	X	X	X
loop	NA	NA	X	X

* Modem should not respond to a break.

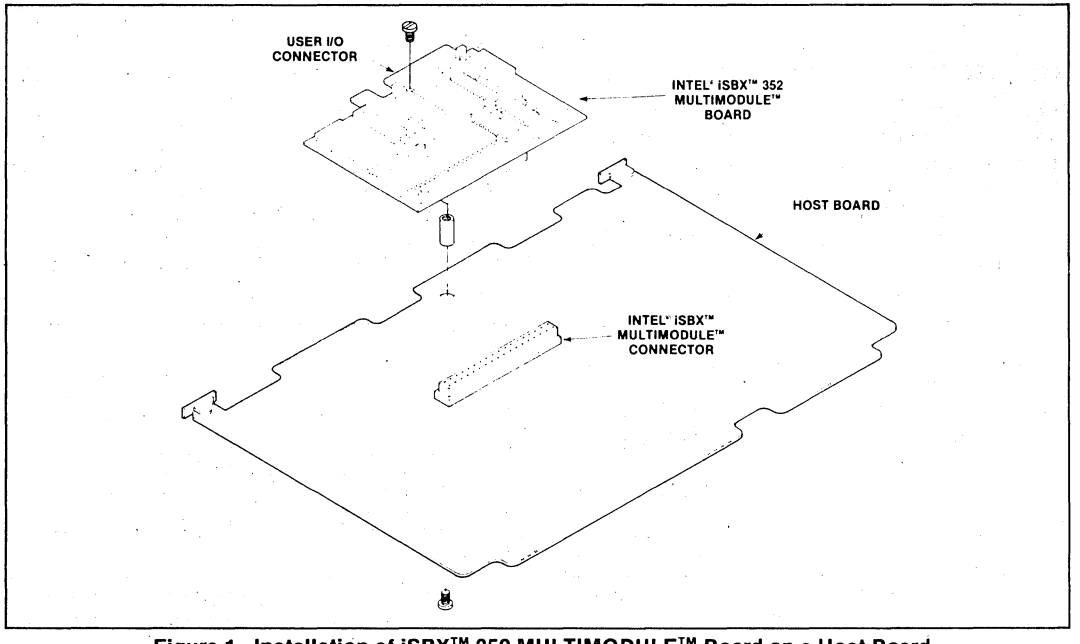


Figure 1. Installation of iSBX™ 352 MULTIMODULE™ Board on a Host Board

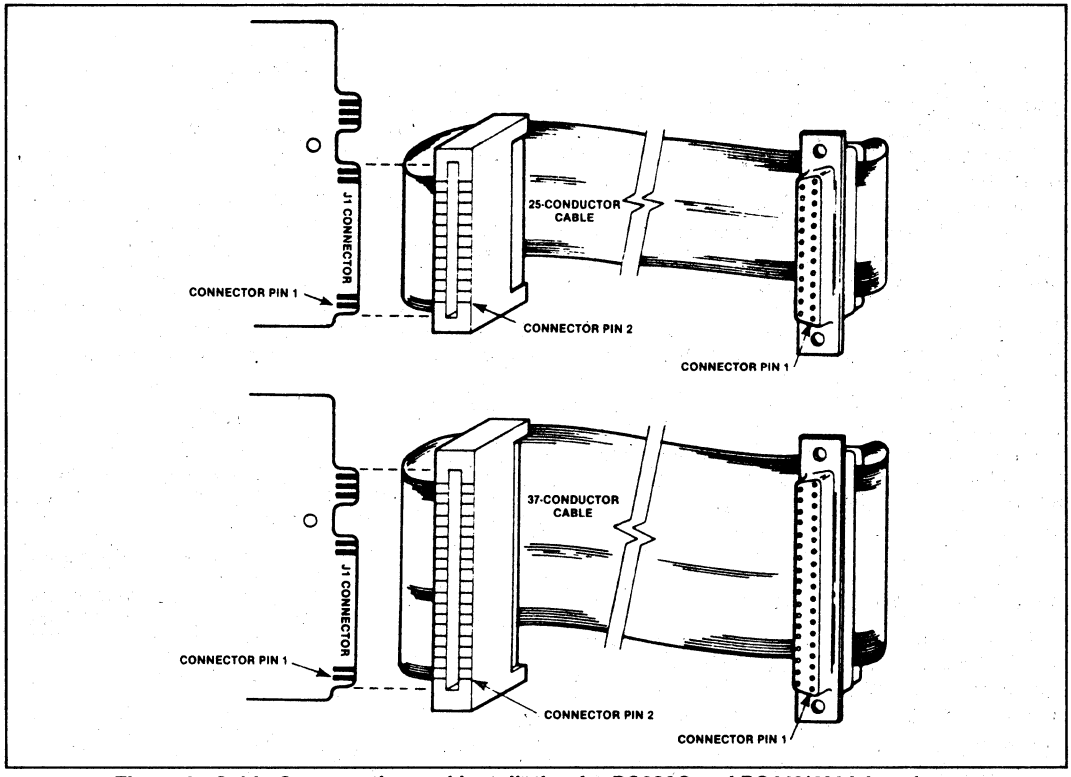


Figure 2. Cable Construction and Installation for RS232C and RS449/422A Interface

SYNCHRONOUS POINT-TO-POINT INTERFACE

Figure 3 shows a synchronous point-to-point mode of operation for the iSBX 352 module. This RS232C example uses a modem for generation of the receive clock for coordination of the data transfer. The iSBX 352 module generates the transmit synchronizing clock for synchronous transmission.

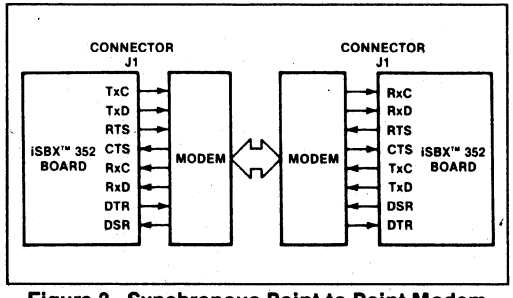


Figure 3. Synchronous Point-to-Point Modem Interface Configuration Example - RS232C

SELF-CLOCKING POINT-TO-POINT INTERFACE

The iSBX 352 module is used in an asynchronous mode interface when configured as shown in Figure 4. The point-to-point RS232C example uses the self-clocking mode interface for NRZI encoding/decoding of data. The digital phase lock loop allows operation of the interface in either half-duplex or full-duplex implementation with or without modems.

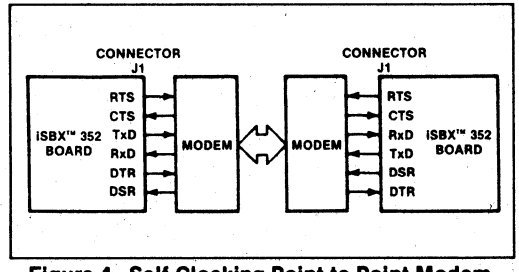


Figure 4. Self-Clocking Point-to-Point Modem Interface Configuration Example - RS232C

SYNCHRONOUS MULTIDROP

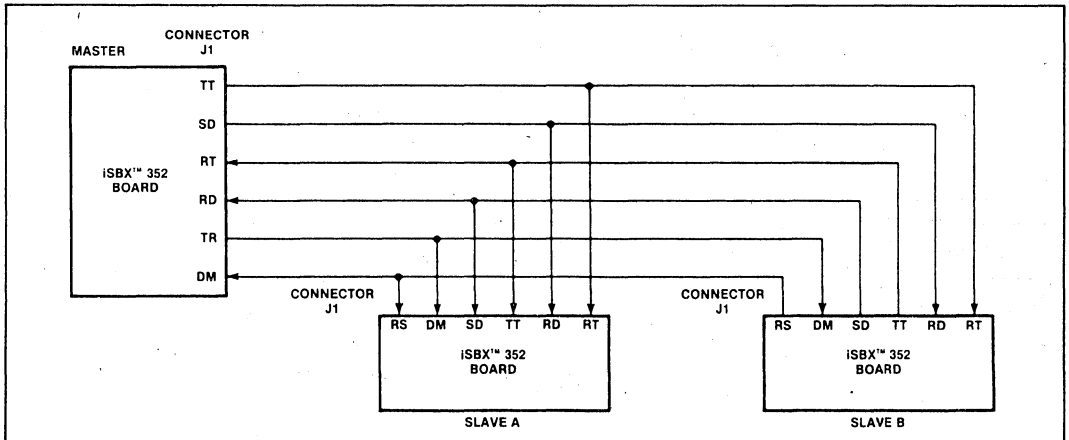
The iSBX 352 MULTIMODULE is used in both a master and a slave mode in the RS449/422A example shown in Figure 5. This synchronous multidrop application is effective for high-speed data transfers between slave stations and a central master station.

ASYNCHRONOUS SELF-CLOCKING MULTIDROP

The iSBX 352 MULTIMODULE example in Figure 6 shows a master and multiple slaves in a multidrop configuration. This self-clocking example uses the 8273 digital phase lock loop and NRZI data encoding.

SDLC Loop

The SDLC self-clocking loop configuration shown in Figure 7 permits longer networks since each secondary slave station is a repeater set in one-bit-delay mode. The data sent out by the primary station (the loop controller) are relayed bit-for-bit through each secondary station and finally back to the master station.



NOTE:
The last slave device in the system must contain termination resistors on all signal lines received by the slave board. The master device must contain resistors on all received signal lines.

Figure 5. Synchronous Multidrop Network Configuration Example - RS422A

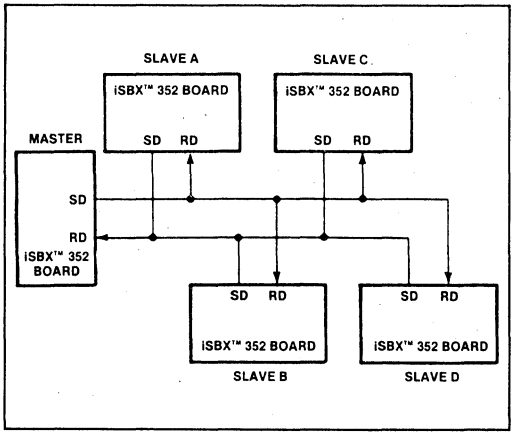


Figure 6. Self-Clocking Multidrop Configuration Example - RS422A

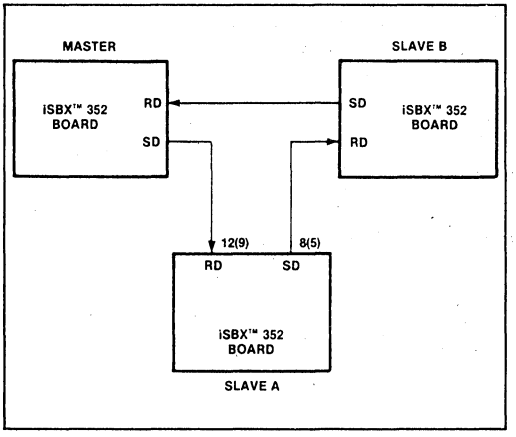


Figure 7. Self-Clocking SDLC Loop Network Configuration Example

SPECIFICATIONS

Data Size

8 Bits

I/O Port Addresses

Port Address		Device Selected	Function Performed
8-bit	16-bit		
X0	X0	8254-2 PIT	Read Counter 0 Write Counter 0
X1	X2		Read Counter 1 Write Counter 1
X2	X4		Read Counter 2 Write Counter 2
X3	X6		Write Control
X4	X8	8273 HDLC/SDLC CONTROLLER	Read Status Write Command
X5	XA		Read Result Write Parameter
X6	XC		Read Transmit Interrupt Write Reset
X7	XE		Read Receive Interrupt
Y0	Y0		Read Receive Data
Y4	Y8		Write Transmit Data

NOTE: Refer to the Hardware Reference Manual for your host iSBC™ microcomputer to determine the upper digit (either X or Y) of the MULTIMODULE™ port address.

Interfaces

iSBX™ BUS — All signals TTL compatible

SERIAL RS232C SIGNALS

CTS	Clear to Send
DSR	Data Set Ready
DTE TXC	Transmit Clock
DTR	Data Terminal Ready
FG	Frame Ground
RTS	Request to Send
RXC	Receive Clock
RXD	Receive Data
SG	Signal Ground
TXD	Transmit Data

RATE GENERATOR FREQUENCIES

Baud Rate bits/sec	8254-2 Divide Count		
	Synchronous	Self-Clocking	
64K	125	TX Clock	32X Clock
56K	143	—	—
48K	167	—	—
19.2K	417	—	—
9.6K	833	833	26
4.8K	1,667	1,667	52
2.4K	3,333	3,333	104
1.2K	6,667	6,667	208
0.6K	13,333	13,333	417
0.3K	26,667	26,667	833

NOTE: All numbers are in decimal notation.

SERIAL RS449/422A SIGNALS

CS	Clear to Send
DM	Data Mode
RC	Receive Common
RD	Receive Data
RS	Request to Send
RT	Receive Timing
SC	Send Common
SD	Send Data
SG	Signal Ground
TR	Terminal Ready
TT	Terminal Timing

OPERATING SPEEDS

24 MHz on-board crystal
8 MHz clocking of the 8254-2 PIT
4 MHz clocking of the 8273 Device

DATA THROUGHPUT SPEED

64K baud maximum for half-duplex operation
48K baud for full-duplex operation issuing commands during transmit operations

SERIAL INTERFACE CONNECTORS

Configuration	Mode[2]	MULTIMODULE™ Edge Connector	Cable	Connector
RS232C	DTE	26-pin[5], 3M-3462-0001	3M[3]-3349/25	25-pin[7], 3M-3482-1000
RS232C	DCE	26-pin[5], 3M-3462-0001	3M[3]-3349/25	25-pin[7], 3M-3483-1000
RS449	DTE	40-pin[6], 3M-3464-0001	3M[4]-3349/37	37-pin[1], 3M-3502-1000
RS449	DCE	40-pin[6], 3M-3464-0001	3M[4]-3349/37	37-pin[1], 3M-3503-1000

NOTES:
1. Cable housing 3M-3485-4000 may be used with the connector.
2. DTE - Data Terminal Equipment mode (male connector); DCE - Data Set Equipment mode (female connector).
3. Cable is tapered at one end to fit the 3M-3462 connector.
4. Cable is tapered to fit 3M-3464 connector.
5. Pin 26 of the edge connector is not connected to the flat cable.
6. Pins 38, 39, and 40 of the edge connector are not connected to the flat cable.
7. May be used with the cable housing 3M-3485-1000.

Electrical Characteristics

DC POWER REQUIREMENTS

Interface	Voltage	Current (max)	Total Power
RS 232C	+5 ± 0.25V	595 mA	3.8 watts
	−12 ± 0.6V	30 mA	
	+12 ± 0.6V	30 mA	
RS 449/422A	+5 ± 0.25V	775 mA	4.1 watts

Environmental Characteristics

Temperature — 0 - 55°C, free moving air across base board and MULTIMODULE board

Humidity — to 90%, without condensation

Physical Characteristics

Width — 7.27 cm (2.85 inches)

Length — 9.40 cm (3.70 inches)

Height — 1.40 cm (0.56 inches)

Weight — 72 gm (2.53 ounces)

Reference Manual (Not Supplied)

143983 — iSBX 352 Bit Serial Communications MULTIMODULE Board Hardware Reference Manual.

Reference manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Ave., Santa Clara, California 95051.

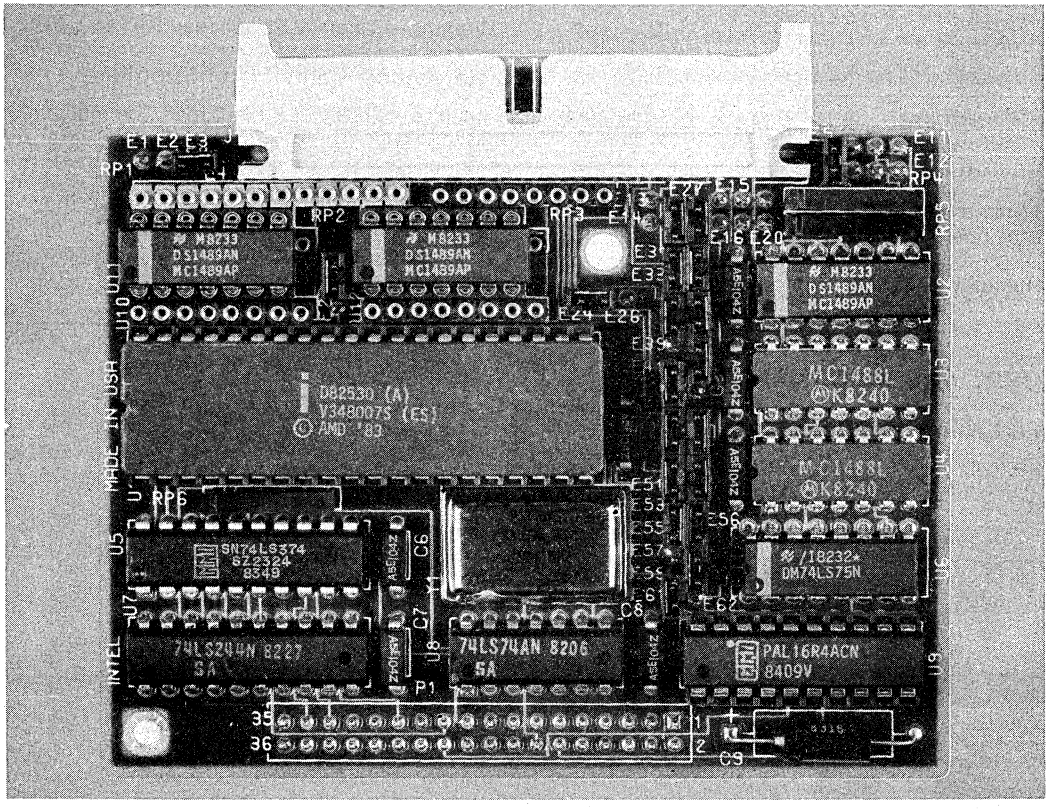
ORDERING INFORMATION

Part Number	Description
SBX 352	HDLC/SDLC Serial I/O MULTIMODULE Board

iSBX™ 354 DUAL CHANNEL SERIAL I/O MULTIMODULE™ BOARD

- **Two RS232C or RS422A/449 programmable Synchronous/ Asynchronous communications channels**
- **Programmable baud rate generation for each channel**
- **Full duplex operation**
- **iSBX™ Bus compatible I/O expansion**
- **Supports HDLC/SDLC, NRZ, NRZI or FM Encoding/Decoding**
- **Three interrupt options for each channel**
- **Low power requirements**

The Intel iSBX™ 354 Serial I/O MULTIMODULE™ board is a member of Intel's line of iSBX compatible MULTIMODULE products. The iSBX MULTIMODULE board plugs directly into any iSBX bus compatible host board offering incremental on-board I/O expansion. Utilizing Intel's 82530 Serial Communications Controller component, the iSBX 354 module provides two RS232C or RS422A/449 programmable synchronous/ asynchronous communications channels. The 82530 component provides two independent full duplex serial channels, on chip crystal oscillator, baud-rate generator and digital phase locked loop capability for each channel. The iSBX board connects to the host board through the iSBX bus. This offers maximum on-board performance and frees the MULTIBUS® System bus for use by other system resources.



JULY 1984
ORDER NUMBER: 280045-001

FUNCTIONAL DESCRIPTION

Communications Interface

The iSBX 354 module uses the Intel 82530 Serial Communications Controller (SCC) component providing two independent full duplex serial channels. The 82530 is a multi-protocol data communications peripheral designed to interface high speed communications lines using Asynchronous, Byte-Synchronous and Bit-Synchronous protocols to Intel's microprocessor based board and system level products. The mode of operation (i.e. asynchronous or synchronous), data format, control character format, and baud-rate generation are all under program control. The 82530 SCC component can generate and check CRC codes in any Synchronous mode and can be programmed to check data integrity in various modes. The command lines, serial data lines, and signal ground lines are brought out to a double edge connector.

The iSBX 354 module provides a low cost means to add two serial channels to iSBC® boards with 8 or 16 bit MULTIMODULE interfaces. In the factory default configuration, the iSBX 354 module will support two RS232C interfaces. With user supplied drivers and termination resistors, the iSBX 354 module can be reconfigured to support RS422A/449 communication interfaces with support on Channel A only for multidrop control from the base board. Both channels can be configured as DTE or DCE with RS232C interfaces.

Interrupt Request Line

The 82530 SCC component provides one interrupt to the MINTRO signal of the iSBX interface. There are six sources of interrupts in the SCC component (Transmit, Receive and External/Status interrupts in both channels). Each type of interrupt is enabled under program control with Channel A having higher priority than Channel B, and with Receive, Transmit and

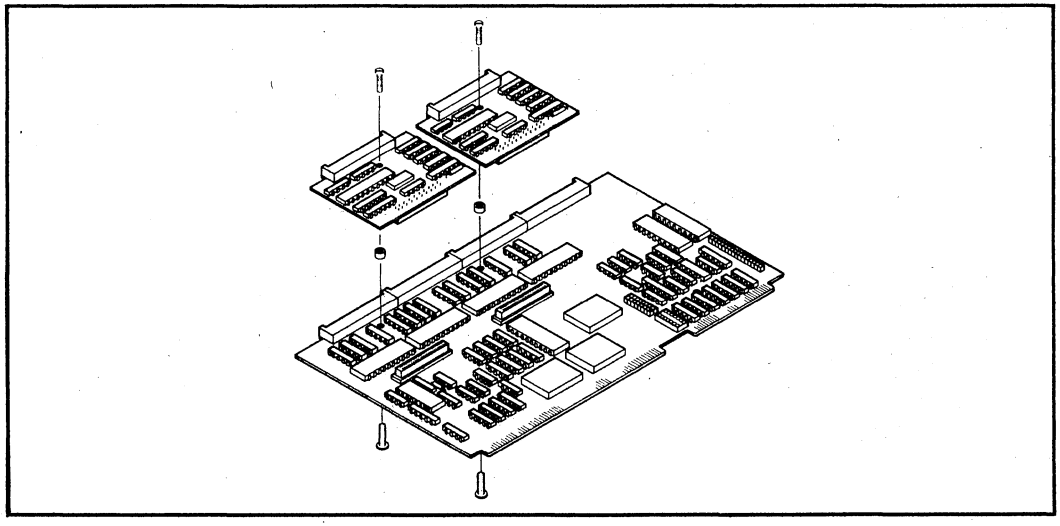


Figure 1. Installation of 2 iSBX™ 354 MULTIMODULE™ Boards on an iSBC® board.

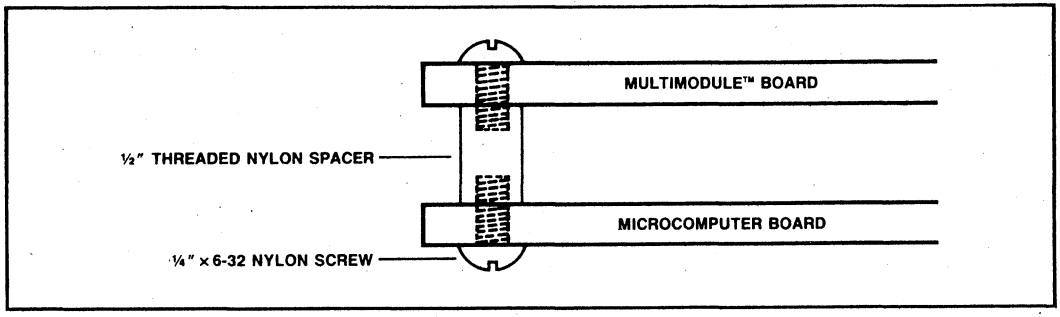


Figure 2. Mounting Technique

External/Status interrupts prioritized in that order within each channel.

Installation

The iSBX 354 module plugs directly into the female iSBX connector on the host board. The module is then secured at one additional point with nylon hardware to insure the mechanical security of the assembly. Figures 1 and 2 demonstrate the installation of the iSBX 354 MULTIMODULE board on a Host Board. Figures 3 and 4 provide cabling diagrams.

Programming Considerations

The Intel 82530 SCC component contains several registers that must be programmed to initialize and control the two channels. Intel's iSBX 354 Module Hardware Reference Manual (Order #146531-001) describes these registers in detail.

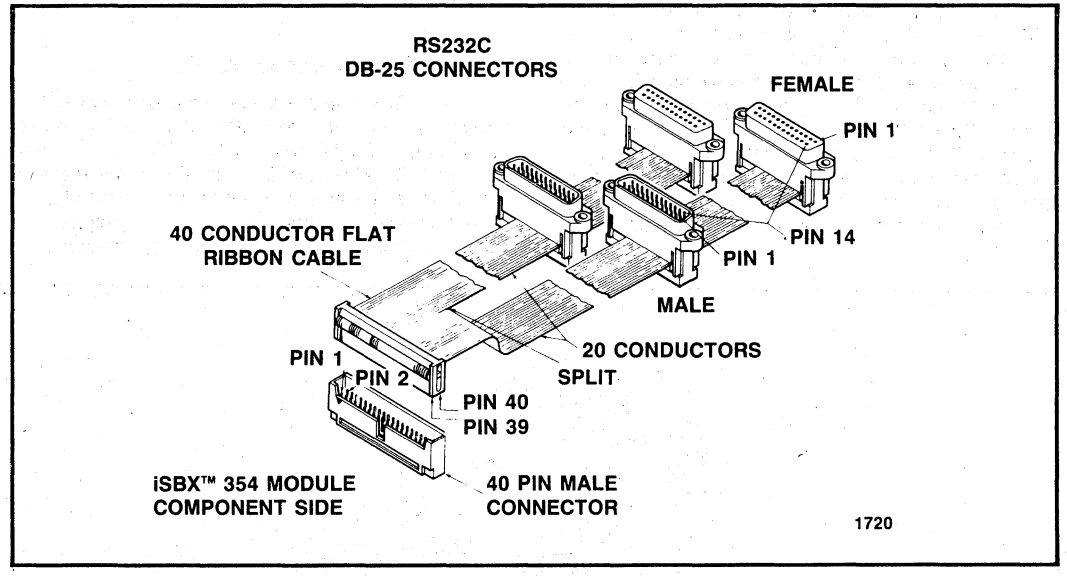


Figure 3. RS232C Cable Construction

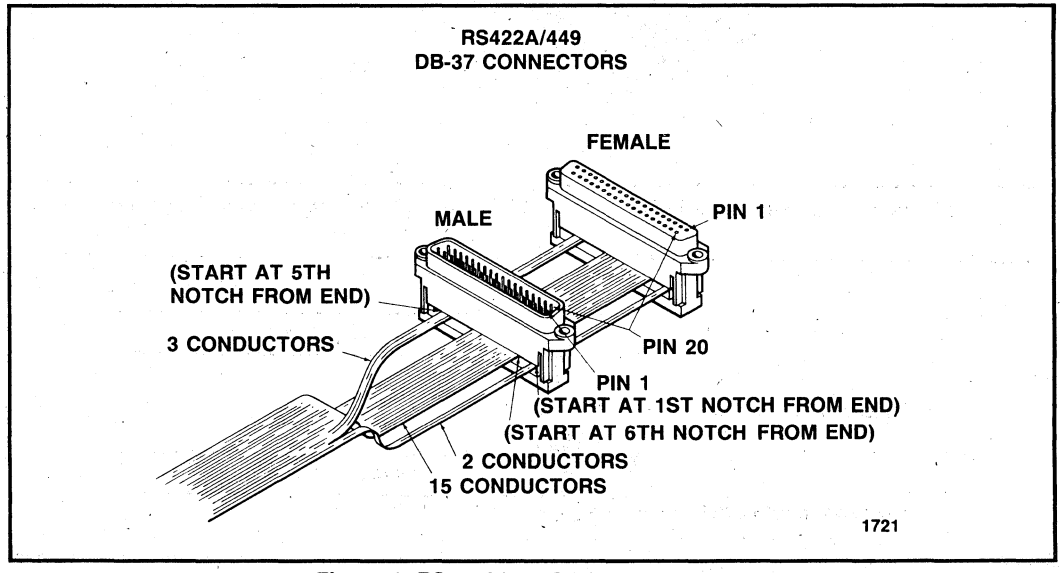


Figure 4. RS422A/449 Cable Construction

SPECIFICATIONS

Word Size

Data — 8 Bits

Clock Frequency

4.9152 MHz

Serial Communications

Synchronous — Internal or external character synchronization on one or two synchronous characters

Asynchronous — 5-8 bits and 1, 1½ or 2 stop bits per character; programmable clock factor; break detection and generation; parity, overrun, and framing error detection

Sample Baud Rate:

Synchronous X1 Clock	
Baud Rate	**82530 Count Value (Decimal)**
64000	36
48000	49
19200	126
9600	254
4800	510
2400	1022
1800	1363
1200	2046
300	8190
Asynchronous X 16 Clock	
Baud Rate	**82530 Count Value (Decimal)**
19200	6
9600	14
4800	30
2400	62
1800	83
1200	126
300	510
110	1394

INTERFACES

iSBX™ Bus: Meets the iSBX Specification, Compliance Level: D8 I

Serial: Meets the EIA RS232C standard on Channels A and B. Meets the EIA RS422A/449 standard on Channels A and B, Multi-drop capability on Channel A only.

Signals Provided

RS232C DTE

- Transmit Data
- Receive Data
- Request to Send
- Clear to Send
- Data Set Ready
- Signal Ground
- Carrier Detect
- Transmit Clock (2)
- Receive Clock
- Data Terminal Ready
- Ring Indicator

RS232C DCE

- Transmit Data
- Receive Data
- Clear to Send
- Data Set Ready
- Signal Ground
- Carrier Detect
- Transmit Clock (2)
- Receive Clock
- Ring Indicator

RS422A/449

- Send Data
- Receive Timing
- Receive Data
- Terminal Timing
- Receive Common

I/O Port Addresses

Port Address		Function
8-Bit	16-Bit	
X0		Read Status Channel B Write Command Channel B
X2		Read Data Channel B Write Data Channel B
X4		Read Status Channel A Write Command Channel A
X6		Read Data Channel A Write Data Channel A
Y0		Read Disable RS422A/449 Buffer Write Enable RS422 A/449 Buffer

NOTES:

1. The "X" and "Y" values depend on the address of the iSBX interface as viewed by the base board.
2. "X" corresponds with Activation of the MCS0/interface signal; "Y" corresponds with Activation of the MCS1/ interface signal.

Power Requirements

+5 Volts at .5 Amps
+12 Volts at 50 mA
-12 Volts at 50 mA

Physical Characteristics:

Width: 2.85 inches

Length: 3.70 inches

Height: 0.8 inches

Weight: 85 grams

Environmental Characteristics:

Temperature: 0° to 55° C operating at 200 linear feet per minute across baseboard and MULTIMODULE board

Humidity: To 90%, without condensation

Reference Manual

146531-001 — iSBX 354 Channel Serial I/O Board Hardware Reference Manual

Reference manuals may be ordered from any Intel sales representative, distributor office, or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, CA 95051.

Ordering Information:

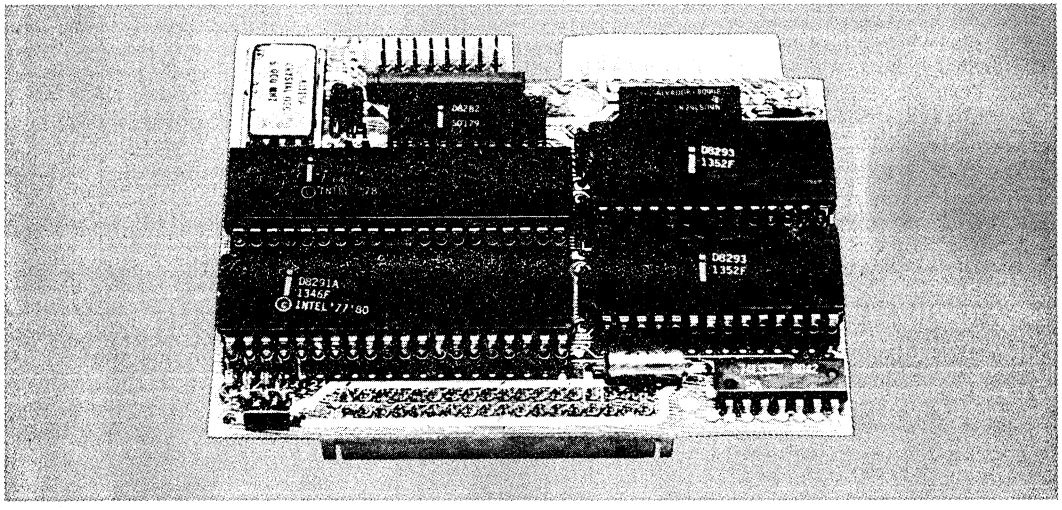
Part Number	Description
iSBX 354	Dual Channel I/O MULTIMODULE

iSBX™ 488
GPIB MULTIMODULE BOARD

- **Complete IEEE 488-1978 talker/listener functions including:**
  - **— Addressing, handshake protocol, service request, serial and parallel polling schemes**
- **Complete IEEE 488-1978 controller functions including:**
  - **— Transfer control, service requests and remote enable**
- **Simple read/write programming**
- **Software functions built into VLSI hardware for high performance, low cost and small size**
- **Standard iSBX™ Bus interface for easy connection to Intel iSBC™ boards**
- **IEEE 488-1978 standard electrical interface transceivers**
- **Five volt only operation**

The intel iSBX™ 488 GPIB Talker/Listener/Controller MULTIMODULE™ board provides a standard interface from any Intel iSBC board equipped with an iSBX connector to over 600 instruments and computer peripherals that use the IEEE 488-1978 General Purpose Interface Bus. By taking full advantage of Intel's VLSI technology the single-wide iSBX 488 MULTIMODULE board implements the complete IEEE 488-1978 Standard Digital Interface for Programmable Instrumentation on a single low cost board. The iSBX 488 MULTIMODULE board includes the 8291A GPIB Talker/Listener, 8292 GPIB Controller and two 8293 GPIB Transceiver devices. This board represents a significant step forward in joining microcomputers and instrumentation using industry standards such as the MULTIBUS® system bus, iSBX bus and IEEE 488-1978. The high performance iSBX 488 MULTIMODULE board mounts easily on Intel iSBX bus compatible single board computers.

A simple user programming interface for easy reading, writing and monitoring of all GPIB functions is provided. This intelligent interface minimizes the impact on host processor bandwidth.



April 1981
143580

FUNCTIONAL DESCRIPTION

The iSBX 488 Multimodule board is a single-wide iSBX bus compatible I/O expansion board that provides a complete implementation of the IEEE 488-1978 Standard Digital Interface for Programmable Instrumentation. The iSBX 488 Multimodule board may be configured to be a GPIB controller, talker, listener or talker/listener. The hardware implementation of the iSBX 488 board takes full advantage of Intel's VLSI capability by using the Intel 8292 GPIB controller, 8291A talker/listener and two (2) 8293 bus transceivers. All communication between the host iSBC board and the iSBX 488 Multimodule board is executed via the Intel standard iSBX connector. Many of the functions that previously were performed by user software have been incorporated into VLSI hardware for high performance and simple programming. Both the Intel 8291A GPIB Talker/Listener device and the 8292 device can each communicate independently with the host processor on the iSBC board depending on configuration. Communication from the host iSBC board to either device on the iSBX 488 board is flexible and may be either interrupt or poll driven depending on user requirements. Data transfers to or from the GPIB may be executed by the host processor's I/O Read and I/O Write commands or with DMA handshaking techniques for very high speed transfers.

GPIB Talker/Listener Capabilities

The Intel 8291A device on the iSBX 488 Multimodule board handles all talker/listener communications between the host iSBC processor board and the GPIB. Its capabilities include data transfer, bus handshake protocol, talker/listener addressing procedures, device clearing and triggering, service requests, and both serial and parallel polling schemes. In executing most procedures the iSBX 488 board does not interrupt the microprocessor on the iSBC processor board unless a byte of data is waiting on input or a byte is sent to an empty output buffer, thus offloading the host CPU of GPIB overhead chores.

SIMPLE PROGRAMMING INTERFACE — The GPIB talker/listener functions can be easily programmed using the high level commands made available by the Intel 8291A on the iSBX 488 Multimodule board. The 8291A device architecture includes eight registers for input and eight registers for output. One each of these read and write registers is used for direct data transfers. The remaining write registers are used by the pro-

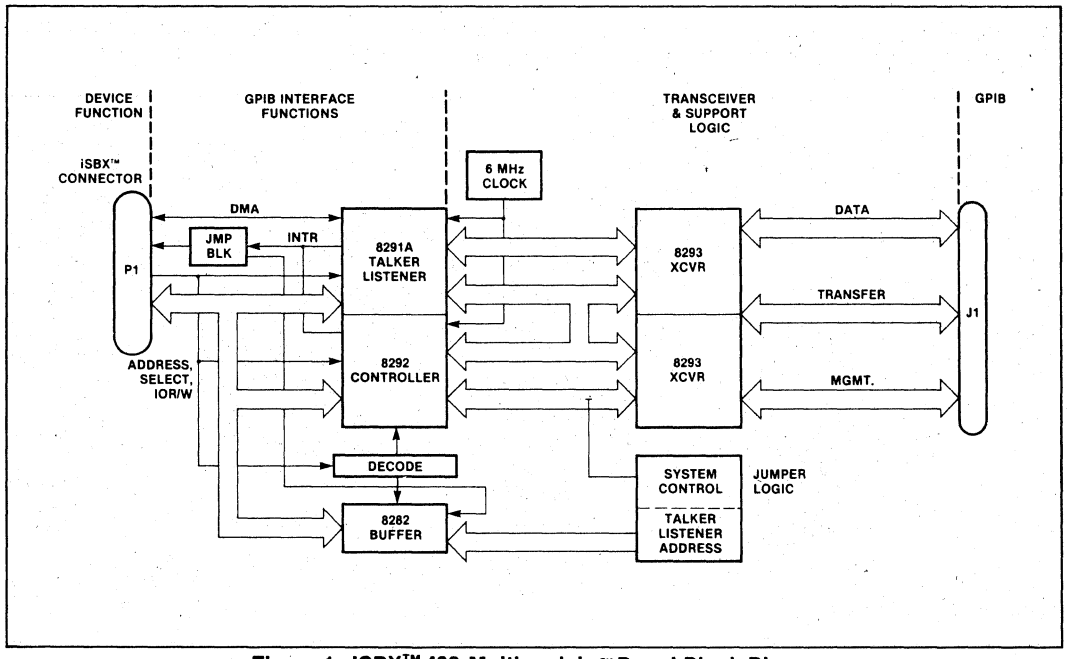


Figure 1. iSBX™ 488 Multimodule ™ Board Block Diagram

grammer to control the various interface features of the Intel 8291A device. The remaining read registers provide the user with a monitor of GPIB states, bus conditions and device status.

SOFTWARE FUNCTIONS BUILT INTO VLSI HARDWARE — Additional features that have migrated from discrete logic and software into Intel VLSI include programmable data transfer rate and three addressing modes that allow the iSBX board to be addressed as either a major or a minor talker/listener with primary or secondary addressing. The iSBX 488 Multimodule board can be programmatically configured into almost any bus talker, listener, or talker/listener configuration. Writing software to control these and other iSBX 488 board functions is simply a matter of reading or writing the control registers.

IEEE 488-1978 Functions[1]

Function	iSBX™ 488 Supported IEEE Subsets
Source Handshake (SH)	SH0, SH1
Acceptor Handshake (AH)	AH0, AH1
Talker (T)	T0 through T8
Extended Talker (TE)	TE0 through TE8
Listener (L)	L0 through L4
Extended Listener (LE)	LE0 through LE4
Service Request (SR)	SR0, SR1
Remote Local (RL)	RL0, RL1
Parallel Poll (PP)	PP0, PP1, PP2
Device Clear (DC)	DC0 through DC2
Device Trigger (DT)	DT0, DT1
Controller (C)	C0 through C28

[1] For detailed information refer to IEEE Standard Digital Interface for Programmable Instrumentation published by The Institute of Electrical and Electronics Engineers, Inc., 1978.

Controller Capabilities

The GPIB controller functions supplied by the iSBX 488 board are provided by the Intel 8292 GPIB controller device. The 8292 is actually an Intel 8041A eight bit microcomputer that has been preprogrammed to implement all IEEE 488-1978 controller functions. The internal RAM in the 8041A is used as a special purpose register bank for the 8292 GPIB Controller. Just as with the 8291A GPIB Talker/Listener device, these registers are used by the programmer to implement controller monitor, read and write commands on the GPIB.

When configured as a bus controller the iSBX 488 board will respond to Service Requests (SRQ) and will issue Serial Polls. Parallel Polls are also issued to multiple GPIB instrument devices for receiving simultaneous responses. In applications requiring multiple bus controllers, several iSBX 488 boards may each be configured as a controller and pass the active control amongst each other. An iSBX 488 board configured for a System Controller has the capability to send Remote Enable (REN) and Interface Clear (IFC) for initializing the bus to a known state.

GPIB Physical Interface

The iSBX 488 Multimodule board interfaces to the GPIB using two Intel 8293 bidirectional transceivers. The iSBX 488 board meets or exceeds all of the electrical specifications defined in IEEE 488-1978 including the required bus termination specifications. In addition, for direct connection to the GPIB, the iSBC 988 cable, a 26 conductor 0.5 meter GPIB interface cable is also available from Intel. The cable is terminated with a 26-pin edge connector at the iSBX end and a 24-pin GPIB connector at the other. The cable is also supplied with shield lines for simple grounding in electrically noisy environments.

Installation

The iSBX 488 Multimodule board plugs directly onto the female iSBX connector available on many Intel iSBC boards. The Multimodule board is then secured at one additional point with nylon hardware (supplied) to insure the mechanical security of the assembly.

SPECIFICATIONS

Interface Information

iSBX™ Bus — All signals TTL compatible
26-pin edge connector — Electrical levels compatible with IEEE 488-1978.

Physical Characteristics

Width — 3.70 in (.94 cm)
Length — 2.85 in (7.24 cm)
Height — 0.8 in (2.04 cm)
Weight — 3.1 oz (87.8 gm)

GPIB Data Rate*

300K bytes/sec transfer rate with DMA host iSBC board

50K bytes/sec transfer rate using programmed I/O

730 nsec Data Accept Time

* Data rates are iSBX board maximum. Data rates will vary and can be slower depending on host iSBC board and user software driver.

Electrical Characteristics

DC power requirements —

V_{CC} = +5 Vdc ±5%

I_{CC} = 600 milliamps maximum

GPIB Electrical and Mechanical Specifications

Conforms to IEEE 488-1978 standard electrical levels and mechanical connector standard when purchased with the iSBC 988 GPIB cable.

Environmental Characteristics

Operating Temperature — 0° to 60°C (32° to 140°F)

Relative Humidity — Up to 90% R.H. without condensation.

Reference Manual

143154-001 — iSBX 488 GPIB Multimodule Board Hardware Reference Manual (not supplied).

ORDERING INFORMATION

Part Number	Description
SBX 488	GPIB Multimodule
SBC 988	0.5 meter GPIB cable for iSBX 488 Multimodule Board

intel

APPLICATION NOTE

AP-96

July 1980

Designing iSBX™ MULTIMODULE™ Boards

Stephen Grubb
OEM Microcomputer Systems Applications

AFN-01931A

INTRODUCTION

Intel's single board computers and the MULTIBUS system bus have become de facto industry standards in the microcomputer board market. The speed and capability of the bus coupled with the functionality and performance of the boards have been used to solve a large number of problems. iSBC products are in applications ranging from simple single board relay replacement to sophisticated multi-board business systems supporting large hard disk files. However, even with the range of functionality provided by standard iSBCs and expansion boards, designers have felt the need to design custom MULTIBUS-compatible boards to fit their application. Until the introduction of the iSBX concept, these custom boards had to be implemented using a separate MULTIBUS form factor board.

Intel has recently introduced a new line of board products and a new bus which are destined to become another industry standard because of the niche they fill. The new iSBX MULTIMODULE boards are designed to extend the functional capabilities of single board computers at a much lower cost than previously possible. iSBX MULTIMODULE boards are supported by a new bus — the iSBX bus, which allows the MULTIMODULE boards to be added directly to the on-board microprocessor bus. iSBX MULTIMODULE boards are from 10 to 20 square inches in size, therefore permitting small modular increments to a single board computer's capabilities.

System designers now have the capabilities of using either standard iSBCs or iSBX MULTIMODULE boards, or designing custom MULTIBUS compatible or iSBX MULTIMODULE boards. Cost-effective solutions are easily realized because of this added flexibility.

This application note discusses the iSBX MULTIMODULE concept, currently available MULTIMODULE boards and the iSBCs which support these boards. The iSBX bus interface specifications are discussed next, followed by consideration for designing custom iSBX MULTIMODULE boards. A specific design example using an Intel® 8279 Programmable Keyboard/Display Controller is presented.

The objective of the note is to introduce the reader to the iSBX MULTIMODULE concept for expanding iSBC functionality and to illustrate how a designer can effectively use this concept with either standard or custom iSBX boards.

References to further documentation on the iSBX bus, specific iSBX MULTIMODULE boards and iSBC host boards currently available may be found in the Related Intel Publications section in the front overleaf of this application note.

iSBX™ MULTIMODULE™ BOARD CONCEPT

The iSBX MULTIMODULE board concept was developed to provide the users of Intel single board computers (iSBCs) with a convenient method to incrementally expand the I/O or the computing capabilities of a single board computer. This expansion is done through the use of a new interface called the iSBX bus interface. This interface gives the user the capability of adding I/O mapped functions directly onto the microprocessor bus via plug-in modules that connect to the iSBC board by means of a special iSBX connector. With the use of this new bus interface, it is now possible to expand or add new features to your iSBC system without incurring large costs and long engineering development times.

There are a number of unique advantages to using the iSBX bus interface for system expansion rather than adding a separate expansion board to your system. First, when expansion is required, the user needs only to buy what is required for the application. Second, it is now possible to return to one board solutions for small systems. One board solutions eliminate the need for expensive backplanes and cardcages. Next, the iSBX interface connects directly to the microprocessor or local bus, as opposed to interfacing to the MULTIBUS system bus, therefore I/O expansion does not require system bus cycles. To the CPU, the iSBX board looks like any other on-board I/O device (Figure 1). Address decode logic exists on the iSBC host board for each iSBX connector on the host board.

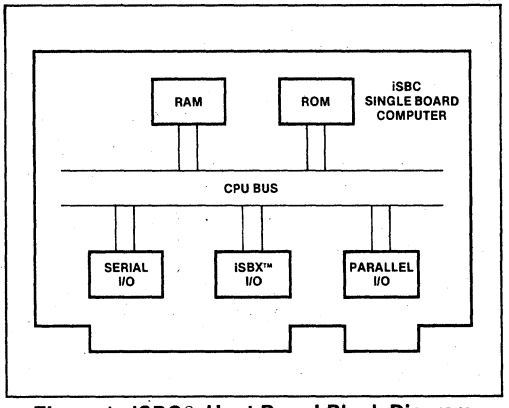


Figure 1. iSBC® Host Board Block Diagram

Third, if there is no iSBC or MULTIBUS compatible expansion board available to fit the needs of your application or if the expansion boards available offer more capability than required, then it is possible to design a custom iSBX MULTIMODULE board. Custom iSBX boards offer several advantages over custom MULTIBUS boards: they require less board real estate (10 or 20

square inches versus 81 square inches) and less engineering design time; consequently, they cost considerably less to implement. Additional capability is therefore achieved with maximum productivity.

Currently available Intel iSBX MULTIMODULE Boards include:

1) iSBX 350 Parallel I/O MULTIMODULE board which contains 24 programmable I/O lines with sockets for line drivers and terminators.
2) iSBX 351 Serial I/O MULTIMODULE board containing one RS232 or RS449/422 programmable synchronous/asynchronous communications channel and two timers.
3) iSBX 331 Fixed/Floating Point Math MULTIMODULE board which permits fixed or floating point mathematics via the Intel 8231 device.
4) iSBX 332 Floating Point Math MULTIMODULE board which permits floating point mathematics using the Intel and proposed IEEE floating point standards via an Intel 8232 device.

With these iSBX MULTIMODULE boards and other soon-to-be-announced boards, the capability now exists to economically tailor a single board computer to the application using off-the-shelf products.

iSBX™ MULTIMODULE™ SYSTEM INTERFACE

This section begins by describing the basic system elements used in an iSBX MULTIMODULE interface configuration and then defines the interface signals used for the communication between these elements. The specifications contained in this application note are included for descriptive and tutorial purposes only. The ultimate source for this information is the iSBX Bus Specification which is referenced in the front overleaf of this note.

Host Boards

The host board provides an electrical and mechanical interface for the iSBX expansion module. The host board is the master of the communications between the host and iSBX board, it controls the address and command signals.

A new generation of iSBX bus compatible host boards are evolving. The first board available from Intel is the iSBC 80/10B Single Board Computer. The 80/10B contains an 8080A CPU operating at 2 MHz, 1K bytes of RAM with sockets available for expansion to 4K bytes of RAM, sockets for up to 16K bytes of EPROM, 24 parallel I/O lines, a programmable synchronous/asynchronous communications interface and a fixed 1.04 msec timer. The 80/10B has one iSBX connector, permitting the use of an iSBX MULTIMODULE board.

The second iSBC board available supporting iSBX boards is the iSBC 80/24 Single Board Computer. The 80/24 board, which supports two iSBX MULTIMODULE boards, contains an 8085A-2 CPU operationg at 4.8 or 2.4 MHz, 4K bytes of RAM, sockets for up to 32K bytes of EPROM, 48 parallel I/O lines, a programmable synchronous/asynchronous communications interface, three programmable interval timers and a programmable interrupt controller. Further RAM expansion on the 80/24 board is accomplished by the addition of an iSBC 301 4K byte RAM MULTIMODULE board which expands the RAM by an additional 4K bytes for a total of 8K bytes. The iSBC 301 MULTIMODULE board is not iSBX bus compatible; it is attached via pins and sockets in the RAM section of the host board.

iSBX™ MULTIMODULE™ Boards

The iSBX MULTIMODULE boards communicate with the host boards via the iSBX bus interface. These iSBX boards are I/O mapped through pre-defined select lines to specific port addresses. The iSBX bus currently defines an 8-bit data path compatible with both 8 and 16-bit future iSBC host boards. Examples of possible iSBX expansion boards include a floppy disk controller, a cassette interface, analog-to-digital converter or digital-to-analog converter boards, an interface to the IEEE 488 Bus and a video graphics display interface board.

There are two standard sizes of iSBX boards: a single-wide board measuring 7.24 by 9.40 cm (2.85 by 3.70 inches) and a double-wide board measuring 7.24 by 19.05 cm (2.85 by 7.50 inches). The iSBX MULTIMODULE boards mount onto any microcomputer board containing an iSBX connector and mounting hole. The iSBX boards physically plug into the iSBX connector on the host board and are secured with a nylon stand-off and screws. The mounting hardware supplied as part of the iSBX board includes:

1) One nylon spacer, 1/2″ threaded
2) Two nylon screws, 1/4″ 6-32
3) One 36-pin connector, factory-installed onto the iSBX module. (These may also be purchased from Intel.)

The interconnection between the host board and iSBX board, as well as the mounting clearances, may be seen in Figures 2 and 3.

NOTE

The iSBX board, when installed onto a host board, occupies an additional card slot adjacent to the base board in an iSBC 604/614 Cardcage. However, the base board may be inserted in the top card slot of the cardcage. If this is done, no additional slots are required.

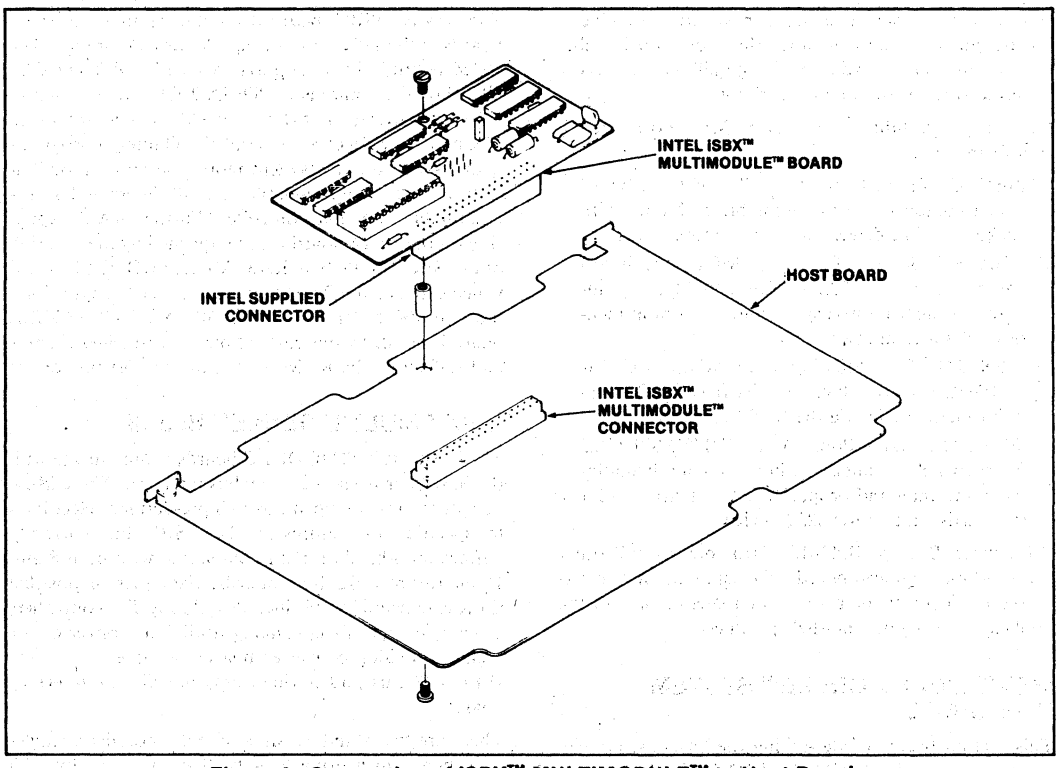


Figure 2. Connection of ISBX™ MULTIMODULE™ to Host Board

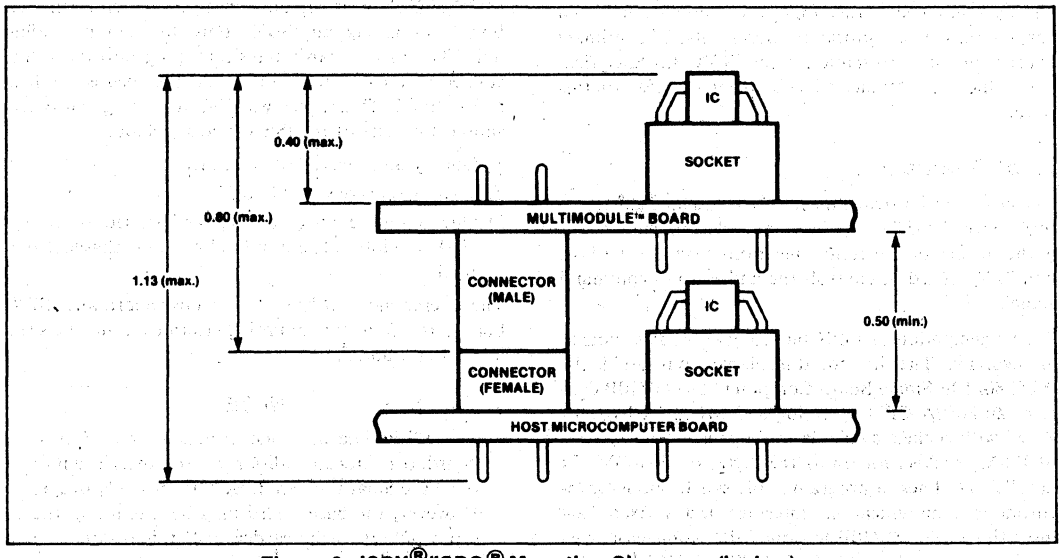


Figure 3. ISBX®/ISBC® Mounting Clearance (Inches)

iSBX™ Connector

The iSBX interface connector is a 36-pin custom made connector that was designed by Intel especially for this interface. The connector is plastic with gold plated contact pins for maximum reliability. The connector for the iSBX interface was designed for high reliability and durability. The connection between the host board and the iSBX MULTIMODULE board was extensively tested for vibration, shock, humidity, and temperature to insure that the connection is rugged enough to be used in severe environments. This connection was tested for the following environment:

Vibration: Sweeping from 10 Hz to 55 Hz and back to 10 Hz at a distance of 0.010 inches peak-to-peak, lasting 15 minutes in each of the three planes.

Shock: 30g's of force for an 11-msec duration, three times in three planes, both sides (total of 18 drops).

Humidity: 90% maximum relative (no condensation).

Temperature: 0 to 55°C (32-131°F) free moving air across the base board and the iSBX MULTIMODULE board.

Further information on the reliability testing that was done on this inter-connection, or reliability information on the iSBX MULTIMODULE boards in general, is contained in the Reliability Report, RR-29, "Intel iSBX MULTIMODULE Boards and iSBC 80/10B Single Board Computer," listed in the overleaf of this note.

The male half of this connector is available from Intel in the form of the iSBX 960-5 package which contains five of the connectors.

iSBX™ Bus Interface Signals

The iSBX bus interface signals are grouped into six basic groups, or classes, according to the functions performed relative to the interface:

These signals are:

CONTROL LINES
ADDRESS LINES
DATA LINES
INTERRUPT LINES
OPTIONAL LINES
POWER LINES

Many of the signals on the iSBX bus are active-low, meaning a low level on a control signal of the bus indicates a logic "1" value, while a low level on an address or data signal of the bus represents a logic "0" value.

NOTE

In this application note, an active-low signal will be designated by placing a slash (/) after the mnemonic for the signal.

Appendix A contains a pin assignment list of the following signals:

CONTROL LINES

The following signals are classified as control lines:

1) COMMANDS — IORD/, IOWRT/
2) DMA — DMRQT, MDACK/, TDMA
3) INITIALIZE — RESET
4) CLOCK — MCLK
5) SYSTEM CONTROL — MWAIT/, MPST/

Command Lines (I/O READ, I/O WRITE)

The command lines are active-low signals which control the communication link between the host board and the iSBX board. An active command line conditioned by chip select indicates to the iSBX board that the address lines are valid and the iSBX board should perform the specified operation.

DMA Lines (MDRQT, MDACK/, TDMA)

The DMA lines control the communication link between the DMA device on the host board and the iSBX module. DMRQT is an active-high output signal from the iSBX board to the host board's DMA device requesting a DMA cycle. MDACK/ is an active-low input signal to the iSBX board from the host board DMA device acknowledging that the requested DMA cycle has been granted. TDMA is used by the iSBC board to terminate DMA activity. The use of the DMA lines is optional as not all host boards will provide DMA channels nor will all iSBX boards be capable of supporting them.

Initialize Line (RESET)

This active-high input line to the iSBX board is generated by the host board to put the iSBX board into a known internal state.

Clock Line (MCLK)

This input line to the iSBX board is a timing signal. The clock frequency is 10 MHz (+0%, −10%), and the clock is asynchronous with respect to all other iSBX bus signals.

System Control Lines (MWAIT/, MPST)

These output signals from the iSBX board control the state of the system. Active MWAIT/ (active-low) will

put the CPU on the host board into a wait state, providing additional time for the iSBX board to perform the requested operation. MPST/ is an active-low signal (usually tied to signal ground) that informs the host board I/O decode logic that an iSBX module has been installed.

ADDRESS AND CHIP SELECT LINES

The address and chip select lines are made up of the following signals:

1) ADDRESS LINES — MA0, MA1, MA2
2) CHIP SELECT LINES — MCS0/, MCS1/

Address Lines (MA0, MA1, MA2)

These active-high input lines to the iSBX boards are generally the least three significant bits of the I/O addresses. In conjunction with the command and chip select lines, they establish the I/O port address being accessed.

Chip Select Lines (MCS0/, MCS1/)

These active-low input lines to the iSBX board are the result of the host board I/O decode logic. When active, the MCS/ lines condition the I/O command signals and thus enable communication between the iSBX board and the host board.

DATA LINES (MD0-MD7)

There are eight bidirectional data lines. These active-high lines are used to transmit or receive information to or from the iSBX ports. MD0 is the least significant bit.

INTERRUPT LINES (MINTR0, MINTR1)

These active-high output lines from the iSBX board are used to make interrupt requests to the host board. These lines are jumper enabled and disabled on the host board via wire wrap posts.

OPTION LINES (OPT0, OPT1)

These two signals are reserved lines that are connected to wire wrap posts on both the host board and the iSBX MULTIMODULE board. They are for unique requirements where a user needs a host board or MULTIBUS bus signal on the iSBX module.

POWER LINES

All host boards provide +5 volts as well as ±12 volts to the iSBX MULTIMODULE board along with signal ground. All power supply voltages are ±5%. Table 1 gives the power supply specifications for the iSBX interface.

Table 1. Power Supply Specifications

Minimum (volts)	Nominal (volts)	Maximum (volts)	Maximum (current)*
+4.75	+5.0	+5.25	3.0A
+11.4	+12	+12.6	1.0A
−12.6	−12	−11.4	1.0A
—	GND	—	6.0A

*Per iSBX MULTIMODULE board mounted on base board.

iSBX™ BUS INTERFACING

This section of the application note focuses on the iSBX interface and design considerations related to interfacing with the iSBX bus. It discusses the way the major operations like READ, WRITE, and DMA work, and the timing diagrams associated with each. There is also a discussion on other considerations for designing with the iSBX bus.

Bus Timing

The AC timing specifications for the iSBX bus interface can be found in Appendix B of this application note. It should be emphasized that the interface timing between the host board and the iSBX MULTIMODULE board is very critical. This is largely due to the fact that the iSBX board is attached directly to the microprocessor bus. If the timing specifications are not met, unpredictable and possibly intermittent operation of the host board may result.

Command Operations

The command lines (IORD/, IOWRT) are driven from the host board by three-state drivers with pull-up resistors or standard TTL totem-pole drivers. These lines indicate to the iSBX board that action is being requested. There are two types of operations for each command line and it is the iSBX board that determines which operation is to be performed.

READ OPERATIONS (IORD/)

Two different types of read operations are possible. The first type of read is called a full speed I/O READ. The host board generates a valid I/O address (MA0-MA2) and a valid chip select signal (MCS1/) which is then sent to the iSBX board; after the set-up times are met, the host board activates the IORD/ line. At this time, the iSBX board must generate valid data from the addressed I/O port in less than 250 ns. The host board then reads the data and removes the READ command, address and chip selects. These are shown in the timing diagram for this operation (Figure 4). The second type of read operation is called an I/O READ with Wait. This READ is used by iSBX boards that cannot perform a full speed read operation. Under this operation the

host board generates the valid address and chip select signals, as in the full speed read. But this time the iSBX board will activate the MWAIT/ signal, which in turn removes the READY input to the CPU, putting it into a Wait state. The CPU, however, first activates the IORD/ signal before going into the Wait state. After valid data is placed on the iSBX data bus by the iSBX board, the iSBX board will remove the MWAIT/ signal. The host board will then read the data and remove the command, address, and chip select lines. This I/O READ with Wait operation is shown in Figure 5.

WRITE OPERATIONS (IOWRT/)

There are also two types of write operations possible: the type performed is again determined by the iSBX board. In the full speed I/O WRITE operation, the host board generates a valid I/O address and chip select and then activates the IOWRT/ line after the necessary set-up times are met. The IOWRT/ line, after being activated, will remain active for 300 ns and the data will be valid for 250 ns before the IOWRT/ command is removed. The host board will then remove the data, address, and chip select lines after the hold times are met, as shown in the timing diagram of this operation (Figure 6).

This second write operation is the I/O WRITE with Wait operation. This WRITE is used by the iSBX boards that cannot write into an I/O port with the full speed write specifications. The host board again generates valid address and chip select signals as in the full speed write operation. However, this time the iSBX board generates the MWAIT/ signal based on address information (chip select and MA0-MA1). The activation of MWAIT/ causes the removal of READY to the CPU, thus causing the CPU to go into a Wait state. The iSBX board removes the MWAIT/ signal (allowing the CPU to leave its Wait state) when it has satisfied the WRITE pulse width requirements. At this time the board removes the WRITE command, followed by the data, address, and chip select lines. This I/O WRITE with Wait operation can be seen in Figure 7.

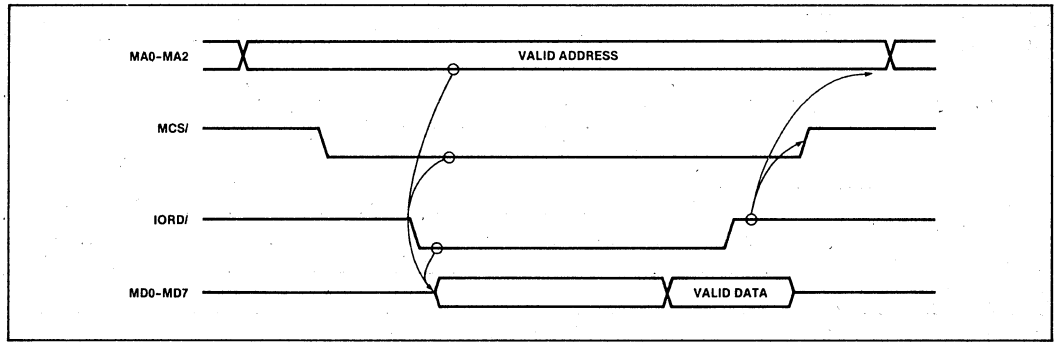


Figure 4. Full Speed I/O Read Operation

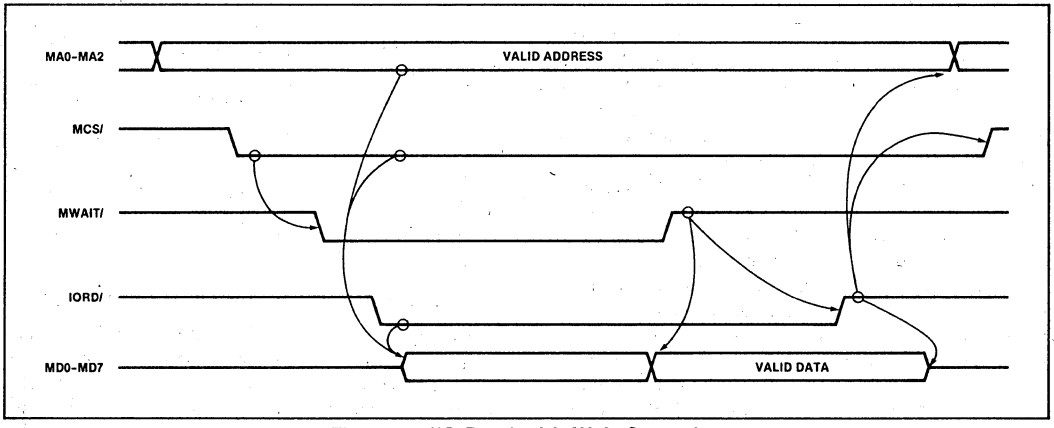


Figure 5. I/O Read with Wait Operation

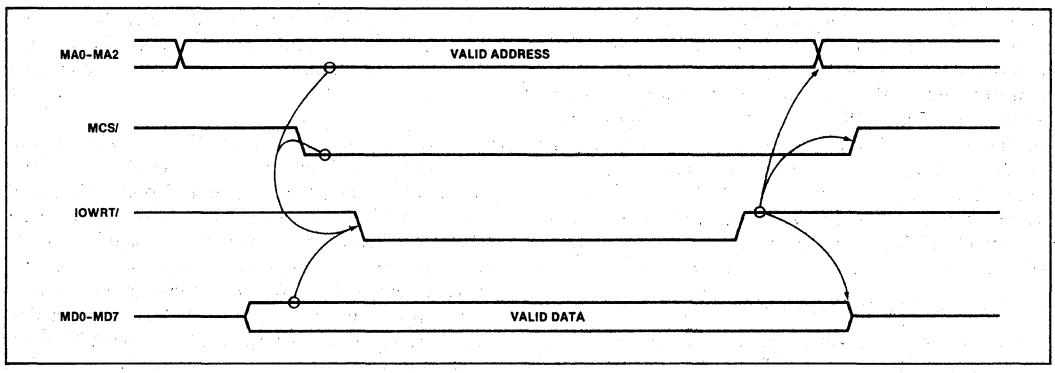


Figure 6. Full Speed I/O Write Operation

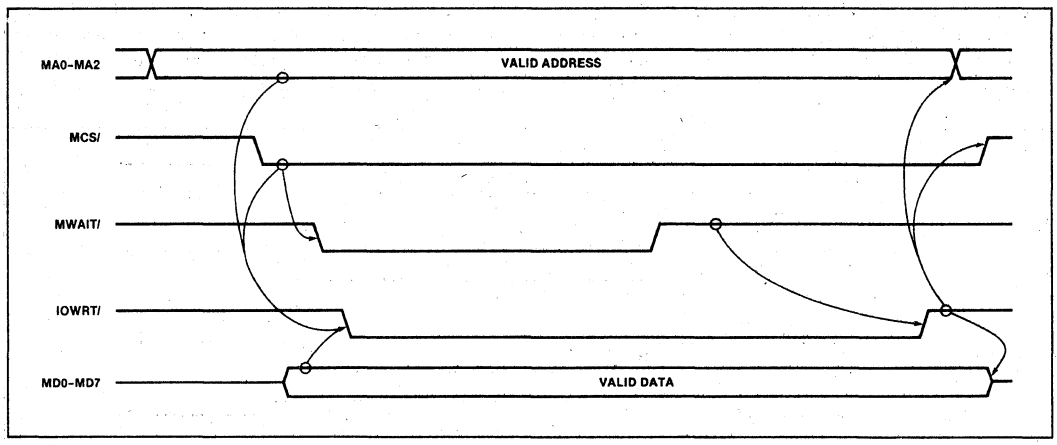


Figure 7. I/O Write with Wait Operation

iSBX™ Addressing

The iSBX boards are addressed by the host board through the use of the address lines MA0, MA1 and MA2, and the chip select lines MCS0/ and MCS1/. The host board decodes the I/O addresses and in turn generates the chip selects for the iSBX boards. In an 8-bit system the host board decodes the high order 13 address bits and generates the appropriate chip select corresponding to those address bits. The low order three address bits are passed to the iSBX board via MA0-MA2. Thus, a host board reserves two blocks of eight I/O ports for each iSBX connector. There can be as many as three iSBX connectors per host board, therefore a total of 48 addresses or six blocks of eight I/O ports that can be reserved for the iSBX boards. Table 2 contains a list of the I/O addresses and their corresponding host board iSBX port assignments of the iSBC 80/10B and iSBC 80/24 host boards.

Table 2. iSBX™ Host Board Port Assignment

iSBX™ Connector Number	Chip Select	iSBX™ Port Addresses
iSBC 80/10B Connector	MCS0/ MCS1/	F0-F7 F8-FF
iSBC 80/24 First Connector	MCS0/ MCS1/	F0-F7 F8-FF
iSBC 80/24 Second Connector	MCS0/ MCS1/	C0-C7 C8-CF

Considerations for iSBX™ Bus Interfacing

When designing with the iSBX interface it is important to note that the iSBX bus is not buffered on the host board. Since there is no isolation between the iSBX board and the host board CPU bus, a short between signal lines and power or ground could have a direct effect

on the CPU or the drivers and receivers associated with the CPU on the host board. This must be taken into consideration, especially when designing and debugging any custom designed iSBX MULTIMODULE board. It is usually during the development states of a product that these types of problems occur. One advantage to not buffering the iSBX bus is increased speed of data and command transfers. Applications requiring buffering may add the buffers on the iSBX board. A second advantage to not buffering is the saving of parts costs, board real estate and development time for the host board. Another consideration when designing with the iSBX interface is, if the application to be designed requires high throughput, like a floppy disk controller or a CRT controller, the designer may consider putting some type intelligent control of buffer RAM onto the iSBX board. By doing this, the transfer information can be stored in this buffer and the throughput of the system increased.

iSBX™ BUS LOADING REQUIREMENTS

Loading requirements for the iSBX bus have been broken up into two basic categories, output specifications and input specifications, which can be viewed in Tables 3 and 4. The output specifications are the requirements on the output drivers of the iSBX board and are the minimum drive requirements necessary. A good example of this would be that the data bus output drivers must be able to sink a minimum of 1.6 mA and maintain V_{OL} at a maximum of 0.5 volts and a minimum source of 200 μA, while providing a minimum output of 2.4 volts. The input specifications are the requirements on the receivers of the iSBX board. An example of this would be that the loading of the address lines (MA0- MA2) can be no greater than 0.5 mA with a minimum low threshold of 0.8 volts.

Optional Interface Lines

The iSBX interface has two optional lines which were included for the user to configure the iSBX board for special application needs. These two lines can be used in a number of ways helpful in unique situations. For example, they could be used as a way to get two extra interrupt lines down to the host board, thus yielding a total of four interrupt lines running between the iSBX MULTIMODULE board and the host board. They could also be used to get extra address lines, or even another clock signal to the iSBX board. They could also

Table 3. Output Specifications

Bus Signal Name	Type[2] Drive	I_{OL} Max – Min (mA)	@ Volts (V_{OL} Max)	I_{OH} Max – Min (μA)	@ Volts (V_{OH} Min)	C_O Min (pF)
MD0-MD7	TRI	1.6	0.5	–200	2.4	130
MINTR0-1	TTL	2.0	0.5	–100	2.4	40
MDRQT	TTL	1.6	0.5	–50	2.4	40
MWAIT/	TTL	1.6	0.5	–50	2.4	40
OPT1-2	TTL	1.6	0.5	–50	2.4	40
MPST/	TTL	Note 3				

Table 4. Input Specifications

Bus Signal Name	Type[2] Receiver	I_{IL} Max (mA)	@ Volts (V_{IN} Max)	I_{IH} Max (μA)	@ Volts (V_{IN} Min)	C_I Max (pF)
MD0-MD7	TRI	–0.5	0.4	70	2.4	40
MA0-MA2	TTL	–0.5	0.4	70	2.4	40
MCS0/-MCS1/	TTL	–4.0	0.4	100	2.4	40
MRESET	TTL	–2.1	0.4	100	2.4	40
MDACK/	TTL	–1.0	0.4	100	2.4	40
IORD/ IOWRT/	TTL	–1.0	0.4	100	2.4	40
MCLK	TTL	2.4	0.4	100	2.4	40
OPT1-OPT2	TTL	2.0	0.4	100	2.4	40

NOTES:
1. Per iSBX MULTIMODULE board.
2. TTL = standard totem-pole output. TRI = three-state.
3. iSBX MULTIMODULE board must connect this signal to ground.

be used to send a special status line to or from the iSBX MULTIMODULE board.

iSBX™ MULTIMODULE™ DESIGN EXAMPLE

This section covers the description of a custom iSBX MULTIMODULE board which uses the Intel 8279 Programmable Keyboard/Display Controller. This iSBX board, when added to an iSBC host board, provides an interface to a keyboard and display. A description of the hardware design considerations for breadboarding the hardware is presented. Following this, a software exerciser, useful for debugging the board, is described. A listing for the exerciser is contained in Appendix C.

Since the iSBX MULTIMODULE board was designed using the Intel 8279 Programmable Keyboard/Display Controller, a brief description of the 8279 is presented. The 8279 is a general purpose programmable keyboard and display I/O controller which was designed for use with the Intel microprocessors. The keyboard portion of this device is capable of providing a scanned interface to a 64-contact key matrix. It is also possible to interface to an array of sensors or a strobed keyboard, such as those of the Hall Effect or the ferrite variety. The 8279 provides a variety of keyboard inputs (i.e., 2-key lockout and N-key rollover), and all key entries are debounced and strobed into an 8-character FIFO. The display portion provides the user with a scanned display interface for LED, incandescent, and other popular display technologies. Both numeric and alphanumeric segment displays may be used, as well as simple indicators. The 8279 is used in this iSBX design example to provide an interface of 2-key lockout with key debounce to a 64-character keyboard, and an interface for a 16-character, 18-segment alphanumeric display.

iSBX™ MULTIMODULE™ Board Design

The iSBX board that was designed for this application note contains a total of three IC's, the keyboard/display controller, a flip-flop, and a 3-to-8-line decoder. Figure 8 contains a block diagram of the hardware used in this design example. Figure 9 contains a schematic for the portion of the design example resident on the custom iSBX board.

The design offers the user some flexibility as to the type of display or keyboard to be attached. For example, if the application design was defined to be for a 7-segment, 16-character display (as the 8279 is designed to drive), a 4-to-16-line decoder along with the display drivers could be added to the iSBX board. Another idea would be to include everything except the display drivers and the display on the iSBX board, and to put the dis-

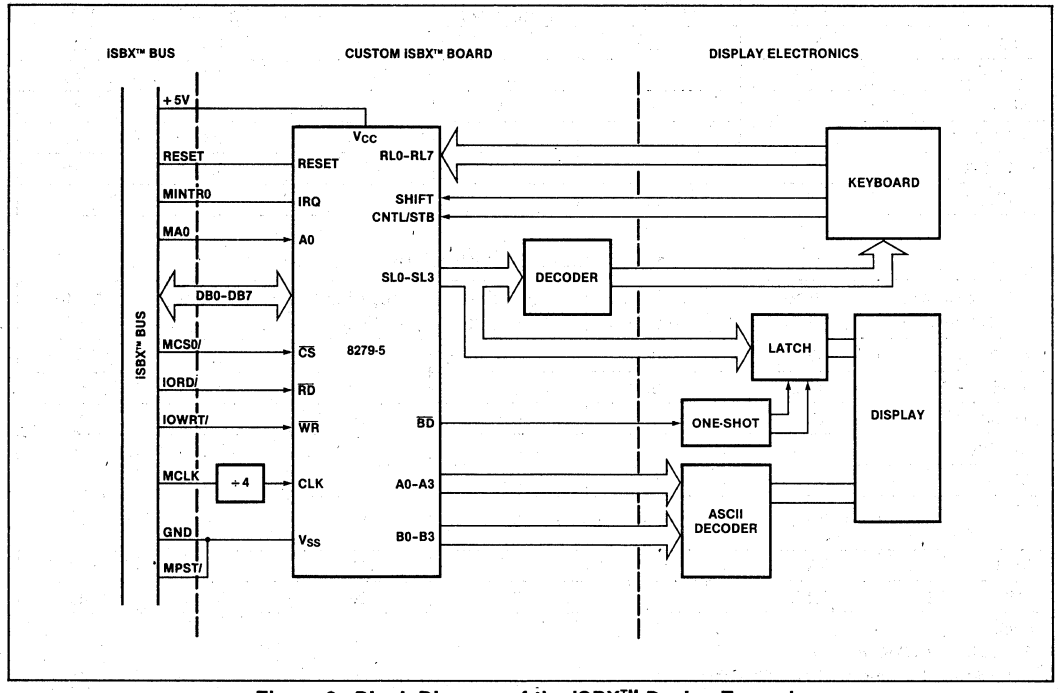


Figure 8. Block Diagram of the iSBX™ Design Example

play and drivers in with the keyboard. It is possible, and probably desirable in some applications, to incorporate some of the display electronics onto the iSBX MULTIMODULE board. Some of the IC's found in the display portion of this design could also have been placed on the iSBX board, as there is enough room on the finished product for doing so.

The design was very easy to implement because, with the exception of one signal, all of the iSBX bus signals necessary to drive the 8279 are connected directly without any extra logic needed. The one signal that would not connect directly to the interface is the clock signal MCLK from the bus to CLK on the controller. It is not possible to connect these two together as MCLK is a 10 MHz signal and the 8279 requires a maximum clock signal of 3.1 MHz to generate its internal timings. It is necessary to add a 74LS74 dual D-type flip-flop to divide the MCLK signal by 4 for the controller. With this exception, all other signals, DB0–DB7 to MD0–MD7, A_0 to MA0, CS/ to MCS0/, etc., are connected directly to the iSBX interface. To meet the timing requirements of the iSBX bus, a high speed version of the 8279, the 8279-5, is used.

The keyboard interface side of the iSBX board consists of a 3-to-8-line decoder, which is used for scanning the keyboard matrix. The 8279 scan lines SL0–SL2 are decoded by a 74LS156 open-collector output decoder and sent to the keyboard via a connector.

The display interface of the iSBX board consists of sending the scan lines and the display outputs to the display module via a connector. The scan lines SL0–SL3 are sent to the display drivers, and the display outputs A0–A3 and B0–B3 are sent to an ASCII to 18-segment decoder driver. The display is discussed in further detail in the next section of this application note.

Display Module Design

The display module design (Figure 10) consists of two 8-digit HDSP 6805 Alphanumeric Displays by Hewlett

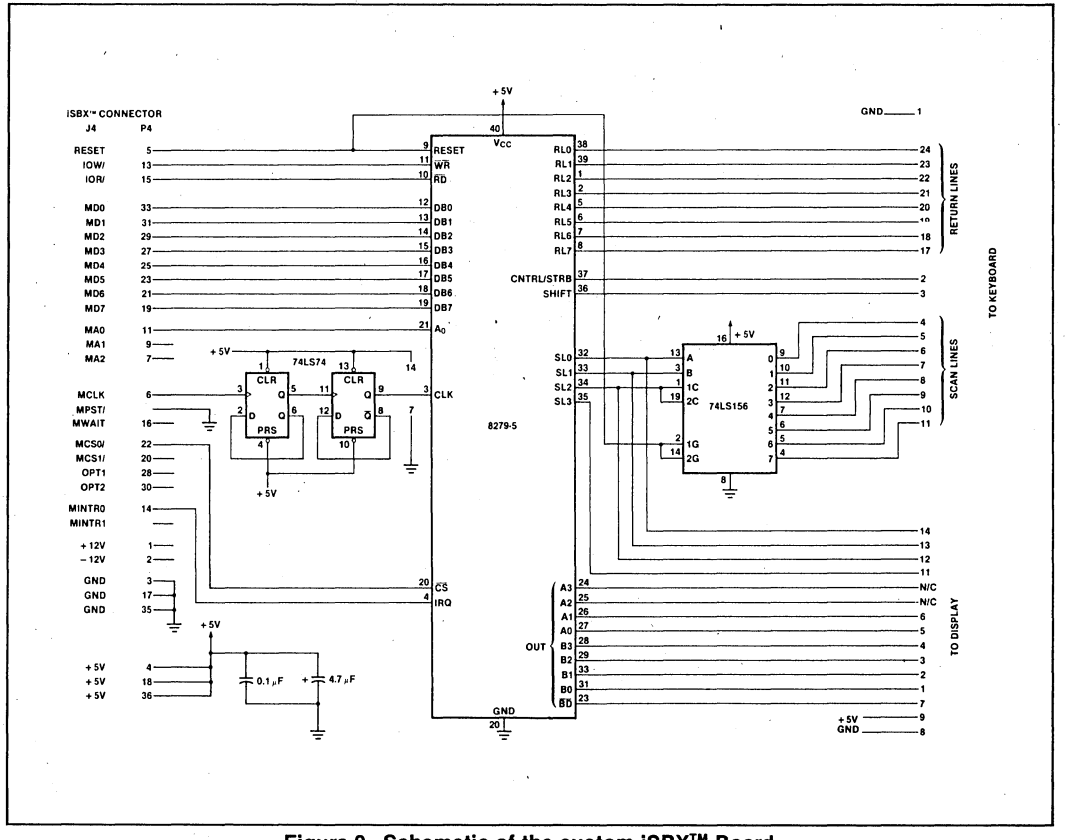


Figure 9. Schematic of the custom iSBX™ Board

Packard, the AC5947 ASCII to 18-segment decoder driver by Texas Instruments, two Signetics NE590 Peripheral Drivers, and a 74LS122 monostable multivibrator. The display is scanned by the outputs A0–A1 and B0–B3, which are connected to the inputs of the AC5947, and the SL0–SL3 outputs which are connected to the NE590 digit scanning circuitry. The interdigit blanking is provided by the 74LS122, which prevents a display ghosting type effect. With the 8279 display controller it is possible for the display to have either left entry, where the data enters from left to right across the display, overflowing in the left most display position, or right entry, where the data enters from the right side of the display and all previous data shifts left. Left entry was chosen for this example. The controller also provides commands for blanking or clearing the display.

Keyboard Interface Design

The eight output lines from the decoder on the iSBX board select 1-of-8 keyboard matrix rows for testing by the controller to see if a key depression has been made in the selected row. The keyboard matrix column output lines are connected directly to the return lines of the 8279, RL0–RL7. Open-collector outputs presented by individual keys within the matrix eliminate the need for isolation diodes when two keys in a given column are depressed. The keyboard/display controller has the option of using either scan keyboard, scan sensor matrix, or strobed input as modes of operation. With the scan keyboard mode there is a choice of using either 2-key lockout or N-key rollover for keyboard entry. The scan keyboard with 2-key lockout mode is used for this ex-

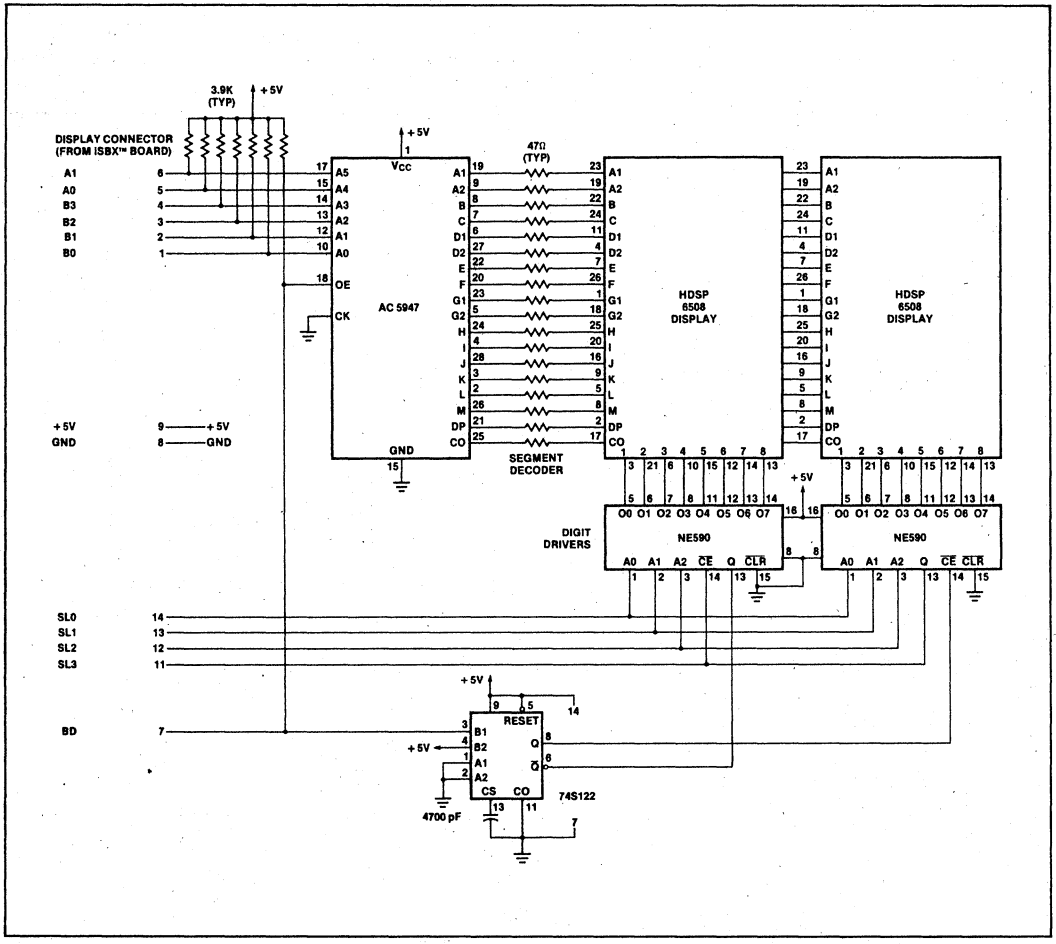


Figure 10. Display Module Schematic

ample. A diagram of the keyboard interfaces and matrix can be seen in Figure 11.

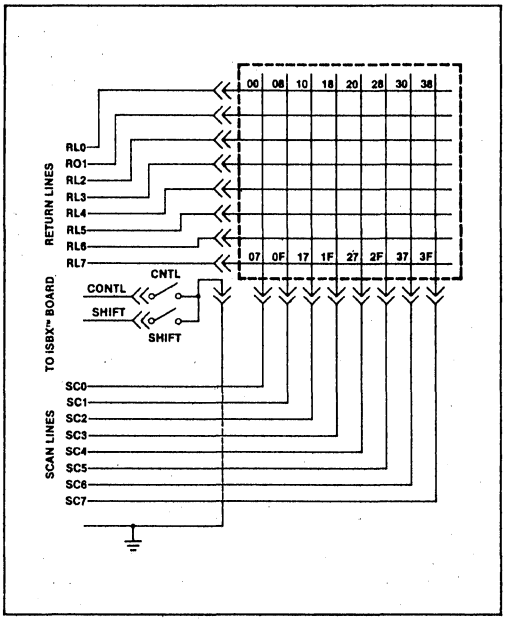


Figure 11. Keyboard Matrix Schematic

Operation with the iSBC®80/10B Single Board Computer

The 8279 on the iSBX expansion board is initialized to its mode of operation following a system reset. The keyboard mode of operation is to scan the keyboard with 2-key lockout, and the display mode is set for the 16-character left entry mode of operation. Upon receiving a character from the keyboard, the 8279 generates an interrupt along the MINTR0 line of the iSBX bus to the CPU. At this time the iSBC 80/10B board commences I/O read operations to the iSBX board by generating valid I/O address and chip select commands on the MA0 and MCS0/ signal lines. After the setup times are met, the 80/10B issues an I/O read command by asserting the IORD/ line on the bus, and the base board reads the data from the iSBX board and removes the IORD/, MA0, and MCS0/ signals from the bus. After the data has been read in from the keyboard, it must be output to the display. The iSBC 80/10B board starts an I/O write operation by generating a valid I/O address and the chip select signal with the MA0 and MCS0/ lines. After the valid setup times are met, the IOWRT/ line is activated by the base board. When the data has been valid for a minimum of 250 ns, the host board removes the IOWRT/ line. When the hold times have been met, the data, address and chip select lines are also removed. Figure 12 shows the timing diagrams just discussed.

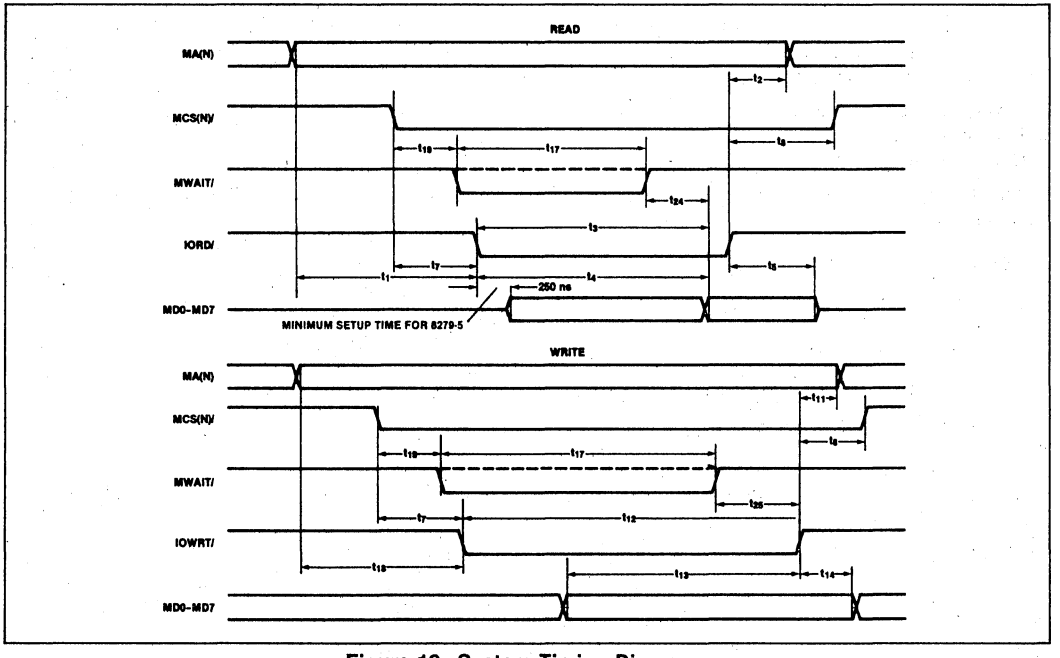


Figure 12. System Timing Diagrams

Breadboarding the Design

When doing the layout of the breadboard, it is also necessary to take into consideration the space required by the mounting holes and to plan the positioning of the components accordingly. (This information is available in the iSBX Bus Specification Manual.)

When attaching the breadboarded design, which typically contains raised wirewrap posts, it is necessary to raise the breadboard well above the host board. This can be accomplished by building a small cable and putting the breadboard on longer nylon standoffs. It is not recommended that the cable be longer than 15 cm (6 in.), otherwise bus timing problems could result.

With the breadboarding finished it is a good idea to recheck all wiring connections for possible errors. Also check all signal lines with an ohmmeter between power, and then ground, for potential shorts. An error at this point can cause serious damage to the host board!

Software Considerations

The software written for this application is an exerciser that is used for hardware checkout. It is a small program designed to echo characters from the keyboard to the display. The software was edited, assembled, linked and located with an Intel development system; it was then debugged with an in-circuit emulator. Both the software and the hardware debug is covered in the next section of this application note.

To facilitate this discussion the software exerciser is divided into three sections based upon the functions performed. The three functions are:

1) Keyboard interrupt routine
2) Initialization and flag checking routine
3) Character output routine

A complete listing of the software exerciser can be found in Appendix C.

KEYBOARD INTERRUPT ROUTINE

The 8279 generates an interrupt to the CPU whenever data is introduced into its FIFO/Sensor RAM. The interrupt is cleared by doing a data read. Whenever a key on the keyboard is depressed an interrupt is generated. Two things are required when an interrupt occurs. First, the keyboard input data must be retrieved and stored. Second, the interrupt routine must indicate that there is some data ready to be output to the display. Therefore, a buffer is created in memory (called "BUFF") at location 3C00H to store the keyboard data. A data present flag is set in a register (REG. C) to indicate that data is ready to be output and can be found in the buffer. In this way the interrupt routine is used to input characters from the keyboard to the input buffer. The buffer is then read by the output routine, which sends the characters to the display.

INITIALIZATION AND FLAG CHECKING ROUTINE

The initialization and flag checking routine first sets the stack pointer to the top of memory. After this the program proceeds to initialize the 8279 Keyboard/Display Controller to its proper mode of operation. The modes of operation used for this application note is scanned keyboard with 2-key lockout for the keyboard, and 16 characters with left entry for the display. As the 8279 has a desired internal operating frequency of 100 kHz, the frequency divider chain is programmed to divide by 19 hex, or 25 decimal. After the 8279 has been initialized, the program begins its next procedure of clearing the buffers. The keyboard input buffer, "BUFF", as well as the display buffer, "DBUFF", are both cleared to a blank display. This is done so that at the time of power up, the display will come up blank. With the initialization now complete, the program disables the interrupts and checks the data present flag for an indication that data might be present for output. If the data present flag is set, the output character routine is called; if it is not set, the interrupts are enabled and the program loops back around to check again. In summary, this routine initializes the 8279 and clears the buffers, and then loops on the data present flag looking for an indication that data is present in the input buffer. The input buffer is a one-byte wide buffer named "BUFF."

CHARACTER OUTPUT ROUTINE

The character output routine brings the character in from "BUFF" (the keyboard input buffer) and compares it to the characters located in a table. If the character can be matched to a character in the table it is replaced in "BUFF" with the corresponding character located in the same position of a second table. If there is no match, it is compared to the code for a control character. If there is no match with a control character, a compare is made to see if the character is a delete charater. When a match is found and the acceptable character is placed in "BUFF", the output routine shifts the data in the display buffer (Figure 13) one position to the left and places the character from the input buffer into the display buffer at position "DBUFF" + 15. Now that

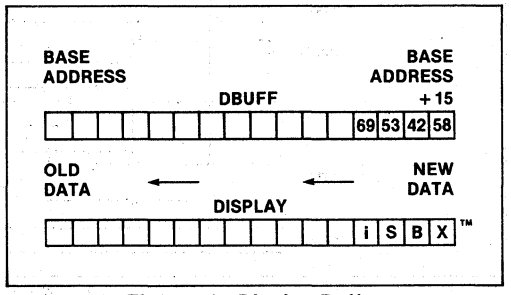


Figure 13. Display Buffer

the new information is in the display, the routine copies the complete contents of the display buffer, "DBUFF", to "DBUFF" + 15 to the display. In the case of the input character being matched up with a delete character, all information in the display buffer is shifted to the right one position and the ASCII code for a blank character is placed into the left-most position or the base address of "DBUFF", thus making the next character sent to the display a blank character. In the case of a control character, nothing is done and the program returns to the flag checking routine.

Debug Considerations

Hardware and software debug was accomplished using an iSBC 80/10B Single Board Computer, an iSBC 655 Chassis, an Intellec® Series II Model 230 Microcomputer Development System, and an ICE-80™ In-Circuit Emulator.

The software was down-loaded from the disk to the iSBC 80/10B board using the in-circuit emulator. The ICE™ module gives the engineer the capability of interrogating the iSBC system by allowing the user to access and display the CPU register contents, status, system memory contents, and all I/O devices and their data.

The iSBC 80/10B board was configured to enable interrupts from the iSBX board via the interrupt 0 line (MINTR0), which is connected to the interrupt pin of the 8080 CPU. The iSBX board was attached to the iSBC 80/10B board via the iSBX connector. The iSBC 80/10B board was powered-up and the iSBX board was checked for proper power and ground connections. The ICE-80 emulator was connected to the iSBC 80/10B board. Using the interrogation mode of the emulator, it is possible to check proper functioning of the iSBX board by sending and receiving data to/from the 8279. The keyboard can be tested by depressing a key on the keyboard and then examining the FIFO/Sensor RAM to see if the data was entered. The display RAM can also be read and written to for testing the interface to the display.

After this initial checking of the iSBX board, the software exerciser can then be down-loaded with the ICE module to further check the board.

SUMMARY

The objective of this application note is to introduce the reader to the iSBX MULTIMODULE concept for expanding a single board computer's functionality, and to illustrate how a designer can use this concept with either standard or custom iSBX boards. In contrast to system expansion using MULTIBUS-compatible boards, iSBX MULTIMODULE boards provide smaller, lower cost, incremental expansion. This application note explains how a custom iSBX board can be designed and debugged. Using this capability, it is now possible to more quickly add new VLSI technology to systems as the technology becomes available. Intel will continue to provide new iSBX MULTIMODULE boards and, because of the the publication of the iSBX Bus Specification and this application note, it will be easier for Intel's customers to also design and build their own custom iSBX boards.

APPENDIX A
iSBX™ SIGNAL PIN ASSIGNMENTS

Pin	Mnemonic	Description	Pin	Mnemonic	Description
35	GND	Signal Ground	36	+5V	+5 Volts
33	MD0	MDATA Bit 0	34	MDRQT	M DMA Request
31	MD1	MDATA Bit 1	32	MDACK/	M DMA Acknowledge
29	MD2	MDATA Bit 2	30	OPT0	Option 0
27	MD3	MDATA Bit 3	28	OPT1	Option 1
25	MD4	MDATA Bit 4	26	TDMA	Terminate DMA
23	MD5	MDATA Bit 5	24		Reserved
21	MD6	MDATA Bit 6	22	MCS0/	M Chip Select 0
19	MD7	MDATA Bit 7	20	MCS1/	M Chip Select 1
17	GND	Signal Ground	18	+5V	+5 Volts
15	IORD/	I/O Read Command	16	MWAIT/	M Wait
13	IOWRT/	I/O Write Command	14	MINTR0	M Interrupt 0
11	MA0	M Address 0	12	MINTR1	M Interrupt 1
9	MA1	M Address 1	10		Reserved
7	MA2	M Address 2	8	MPST/	iSBX MULTIMODULE Board Present
5	RESET	Reset	6	MCLK	M Clock
3	GND	Signal Ground	4	+5V	+5 Volts
1	+12V	+12 Volts	2	−12V	−12 Volts

All undefined pins are reserved for future use.

APPENDIX B
iSBX™ MULTIMODULE™ BOARD I/O AC SPECIFICATIONS

Symbol*	Parameter	Min (ns)	Max (ns)
t_1	Address stable before read	50	—
t_2	Address stable after read	30	—
t_3	Read pulse width	300	—
t_4(2)	Data valid from read	0	250
t_5(2)	Data float after read	0	150
t_6	Time between RD and/or WRT	—	Note 3
t_7	CS stable before CMD	25	—
t_8	CS stable after CMD	30	—
t_9	Power up reset pulse width	50 ms	—
t_{10}	Address stable before WRT	50	—
t_{11}	Address stable after WRT	30	—
t_{12}(2)	Write pulse width	300	—
t_{13}(2)	Data valid to write	250	—
t_{14}	Data valid after write	30	—
t_{15}	MCLK cycle	100	110
t_{16}	MCLK width	35	65
t_{17}(1)	MWAIT/ pulse width	0	4 ms
t_{18}	Reset pulse width	50 ms	—
t_{19}	MCS/ to MWAIT/ valid	0	75
t_{20}	DACK set up to I/O CMD	100	—
t_{21}	DACK hold	30	—
t_{22}	CMD to DMA RQT removed to end of DMA cycle	—	200
t_{23}	TDMA pulse width	500	—
t_{24}(1)	MWAIT/ to valid read data	—	0
t_{25}(1)	MWAIT/ to WRT CMD	0	—

NOTES:

1. Required only if WAIT is activated.
2. If MWAIT/ not activated.
3. To be specified by each iSBX MULTIMODULE board.

* For a more complete definition of symbols refer to iSBX Bus Specification, 142686-001.

APPENDIX C
LISTING FOR THE iSBX™ DESIGN EXAMPLE SOFTWARE EXERCISER

```
LOC  OBJ          LINE        SOURCE STATEMENT

                     1 ;*****************************************************************************
                     2 ;*                                                                           *
                     3 ;*       THIS PROGRAM WAS USED AS AN EXAMPLE FOR EXERCISING THE              *
                     4 ;*       8279 iSBX MULTIMODULE BUILT FOR THIS APPLICATION NOTE.              *
                     5 ;*                                                                           *
                     6 ;*****************************************************************************
                     7
                     8
                     9 ;*****************************************************************************
                    10 ;                        PROGRAM EQUATES
                    11 ;*****************************************************************************
                    12
00F0                13 DATAAD           EQU             0F0H    ; PORT ADDRESS TO READ OR WRITE
                    14                                          ;/DATA TO/OR FROM KEYBOARD/DISPLAY
00F1                15 CMDAD            EQU             0F1H    ; PORT ADDRESS TO SEND COMMANDS
                    16                                          ;/TO KEYBOARD/DISPLAY
0008                17 MODE0            EQU             08H     ; CONTROL CHAR. TO SET
                    18                                          ;/KEYBOARD/DISPLAY MODE FOR
                    19                                          ;/(2 KEY LOCKOUT,16 CHAR LEFT ENTRY
0039                20 PROGCK           EQU             39H     ; CONTROL CHAR. TO SET 8279 CLK
                    21                                          ;/TO 100 KHz INTERNAL TIMING
0040                22 RDFIFO           EQU             40H     ; CONTROL CHAR. TO READ KEYBOARD
0060                23 RDRAM            EQU             60H     ; CONTROL CHAR. TO READ DISPLAY RAM
0070                24 RDRAMA           EQU             70H     ; CONTROL CHAR. TO READ DISPLAY RAM
                    25                                          ;/AUTO INCREMENT
0080                26 WRRAM            EQU             80H     ; CONTL CHAR. TO WRITE TO DISPLAY RAM
0090                27 WRRAMA           EQU             90H     ; CONTL CHAR. TO WRITE TO DISPLAY
                    28                                          ;/RAM AUTO INCREMENT
00D8                29 CLR              EQU             0D8H    ; CONTROL CHAR. TO CLEAR OR BLANK
                    30                                          ;/DISPLAY
3C00                31 BUFF             EQU             3C00H   ; ADDRESS OF KEYBOARD INPUT BUFFER
3D00                32 DBUFF            EQU             3D00H   ; ADDRESS OF DISPLAY BUFFER
                    33
                    34 ;*****************************************************************************
                    35
0000 F3             36 START:           DI
0001 C33B00         37                  JMP     BEGIN
                    38
                    39 ;************************   RST 7 ENTRY POINT   *********************************
                    40
0038                41                  ORG     38H
0038 C3D100         42                  JMP     INT
                    43
                    44 ;*****************************************************************************
                    45 ;                        INITIALIZE PROGRAM
                    46 ;                   AND KEY BOARD DISPLAY CONTROLLER
                    47 ;
003B 31FF3F         48 BEGIN:           LXI     SP,3FFFH        ; INITIALIZE STACK PT
003E 3E08           49                  MVI     A,MODE0         ; GET CONTROL CHAR.
0040 D3F1           50                  OUT     CMDAD           ; SET KEYBOARD/DISPLAY MODE
0042 3E39           51                  MVI     A,PROGCK        ; GET CONTROL CHAR.
0044 D3F1           52                  OUT     CMDAD           ; SET 8279 CLK FOR 100 KHz
0046 3ED8           53                  MVI     A,CLR           ; GET CONTROL CHAR.
0048 D3F1           54                  OUT     CMDAD           ; CLEAR OR BLANK DISPLAY
004A 0EE0           55                  MVI     C,0E0H
004C 21003C         56                  LXI     H,BUFF          ; SET POINTER TO INPUT BUFFER
004F 71             57                  MOV     M,C             ; CLEAR INPUT BUFFER TO BLANK CODE
0050 060F           58                  MVI     B,0FH           ; SET COUNTER = 15
0052 210F3D         59                  LXI     H,DBUFF+0FH     ; SET POINTER TO DBUFF +15
0055 71             60 ZDBUFF:          MOV     M,C             ; CLEAR DISPLAY BUFFER TO
0056 2B             61                  DCX     H               ;/DISPLAY BUFFER +15 TO CODE
0057 05             62                  DCR     B               ;/FOR CLEARING OR BLANKING OUT
0058 C25500         63                  JNZ     ZDBUFF          ;/THE DISPLAY
                    64
                    65 ;*****************************************************************************
                    66 ;                  THIS IS THE BACKGROUND PROGRAM
                    67 ;              WHICH LOOPS CHECKING FOR THE DATA PRESENT FLAG
                    68 ;
005B F3             69 CKFLAG:          DI                      ; DISABLE INTERRUPTS
005C AF             70                  XRA     A               ;/CLEAR A REG AND COMPARE WITH
005D B9             71                  CMP     C               ;/C REG CHECKING FOR DATA PRESENT
005E CA6400         72                  JZ      LABEL           ;/IF PRESENT CALL OUTPT
0061 CD6800         73                  CALL    OUTPT           ;/TO DISPLAY CHAR.
0064 FB             74 LABEL:           EI                      ;/IF NO DATA PRESENT ENABLE
0065 C35B00         75                  JMP     CKFLAG          ;/INTERRUPTS AND JMP BACK
                    76
```

```
                        ;*****************************************************************************
                        ;                       OUTPUT CHARACTER TO DISPLAY
                        ;
0068 3A003C             OUTPT:          LDA     BUFF            ; LOAD A WITH KEYBOARD DATA
006B 062B                               MVI     B,2BH           ; SET COUNTER MAX POSSIBLE CHAR.
006D 21DE00                             LXI     H,TABLE1        ; SET POINTER TO INPUT TABLE
0070 110901                             LXI     D,TABLE2        ; SET POINTER TO OUTPUT TABLE
0073 BE                 COMPARE:        CMP     M               ; COMPARE KEYBD DATA TO INPUT
0074 CA8000                             JZ      MATCH           ;/TABLE IF = JMP TO MATCH
0077 05                                 DCR     B               ;/ELSE DECREMENT COUNTER IF 0
0078 CAC600                             JZ      CONTROL         ;/JMP TO CONTROL
007B 23                                 INX     H               ;/ELSE INCREMENT BOTH TABLE
007C 13                                 INX     D               ;/POINTERS AND JMP TO COMPARE
007D C37300                             JMP     COMPARE
0080 EB                 MATCH:          XCHG                    ; IF MATCH CHANGE INPUT WITH
0081 7E                                 MOV     A,M             ;/OUTPT DATA AND PLACE IN BUFF
0082 21003C                             LXI     H,BUFF
0085 77                                 MOV     M,A
0086 060F                               MVI     B,0FH           ; SET COUNTER = TO 15
0088 11003D                             LXI     D,DBUFF         ; POINTER TO FIRST LOC IN DBUFF
008B 21013D                             LXI     H,DBUFF+1       ; POINTER TO 2ND LOC IN DBUFF
008E 7E                 LOOP1:          MOV     A,M             ; READ HIGH POINTER FROM DBUFF
008F 23                                 INX     H               ;/UPDATE HIGH POINTER
0090 EB                                 XCHG
0091 77                                 MOV     M,A             ; SHIFT DATA LEFT IN D BUFF
0092 23                                 INX     H               ; UPDATE LOW POINTER
0093 EB                                 XCHG
0094 05                                 DCR     B               ; TEST IF DONE
0095 C28E00                             JNZ     LOOP1           ;/AND GO BACK IF NOT
0098 3A003C                             LDA     BUFF            ;/ELSE READ KEYBOARD DATA
009B 320F3D                             STA     DBUFF+0FH       ;/AND PLACE IT IN THE DBUFF
009E 0610               LOOPA:          MVI     B,10H           ; SET COUNTER = 16
00A0 21003D                             LXI     H,DBUFF         ; SET POINTER = DBUFF 1ST POS.
00A3 7E                 LOOP2:          MOV     A,M             ;/READ 1 BYTE FROM DBUFF
00A4 D3F0                               OUT     DATAAD          ;/AND SENT IT TO DISPLAY
00A6 23                                 INX     H               ; UPDATE POINTER
00A7 05                                 DCR     B               ;/AND TEST IF DONE
00A8 C2A300                             JNZ     LOOP2           ;/GO BACK IF NOT DONE
00AB 0E00                               MVI     C,0H            ;/ELSE CLR DATA PRESENT FLAG
00AD C9                                 RET                     ;/AND RETURN
                        ;*****************************************************************************
                        ;                           CHARACTER DELETE
                        ;                             OR RUB OUT
                        ;
00AE 060F               DELETE:         MVI     B,0FH           ; SET COUNTER =15
00B0 110F3D                             LXI     D,DBUFF+0FH     ; SET POINTER = DBUFF+15
00B3 210E3D                             LXI     H,DBUFF+0EH     ; SET POINTER = DBUFF+14
00B6 7E                 LOOPB:          MOV     A,M             ; READ LOW POINTER FROM DBUFF
00B7 2B                                 DCX     H               ;/UPDATE LOW POINTER
00B8 EB                                 XCHG
00B9 77                                 MOV     M,A             ; SHIFT DATA RIGHT IN DBUFF
00BA 2B                                 DCX     H               ;/UPDATE HIGH POINTER
00BB EB                                 XCHG
00BC 05                                 DCR     B               ; TEST IF DONE
00BD C2B600                             JNZ     LOOPB           ;/AND GO BACK IF NOT
00C0 EB                                 XCHG                    ;/ELSE SET DBUFF FOR
00C1 36E0                               MVI     M,0E0H          ;/CODE TO BLANK DISPLAY
00C3 C39E00                             JMP     LOOPA           ;/AND JMP TO LOOPA
                        ;*****************************************************************************
                        ;                       CHECK IF CHARACTER IS
                        ;                   A CONTROL OR DELETE CHARACTER
                        ;
00C6 FEFA               CONTROL:        CPI     0FAH            ; COMPARE FOR CONTROL CHAR.
00C8 CA3B00                             JZ      BEGIN           ;/IF CONTROL JMP TO BEGIN
00CB FEF9                               CPI     0F9H            ;/ELSE COMP. FOR DELETE CHAR.
00CD CAAE00                             JZ      DELETE          ;/IF DELETE JMP TO DELETE
00D0 C9                                 RET                     ;/ELSE RETURN
                        ;*****************************************************************************
                        ;                         KEYBOARD INPUT
                        ;                       INTERRUPT ROUTINE
                        ;
00D1 3E40               INT:            MVI     A,RDFIFO        ; GET CONTL CHAR. TO READ FIFO
00D3 D3F1                               OUT     CMDAD           ; SET 8279 FOR READ MODE
00D5 DBF0                               IN      DATAAD          ; READ KEYBOARD DATA IN
00D7 21003C                             LXI     H,BUFF          ; SET POINTER TO BUFF
00DA 77                                 MOV     M,A             ;/AND STORE KEYBOARD DATA
00DB 0EFF                               MVI     C,0FFH          ;/THEN SET DATA PRESENT FLAG
00DD C9                                 RET                     ;/AND RETURN
```

```
                        158 ;*****************************************************************************
                        159 ;                                   TABLE 1
                        160 ;                ACCEPTABLE INPUT CHARACTERS FROM KEYBOARD
                        161 ;
00DE DE                 162 TABLE1:         DB      0DEH,0FFH,0EFH,0EEH,0E5H,0F6H,0FEH,0C6H
00DF FF
00E0 EF
00E1 EE
00E2 E5
00E3 F6
00E4 FE
00E5 C6
00E6 C9                 163                 DB      0C9H,0CAH,0D2H,0DAH,0D3H,0C7H,0D1H,0D9H
00E7 CA
00E8 D2
00E9 DA
00EA D3
00EB C7
00EC D1
00ED D9
00EE D5                 164                 DB      0D5H,0EDH,0E6H,0F5H,0C1H,0F7H,0DDH,0E7H
00EF ED
00F0 E6
00F1 F5
00F2 C1
00F3 F7
00F4 DD
00F5 E7
00F6 FD                 165                 DB      0FDH,0DFH,0CCH,0D4H,0DCH,0E4H,0ECH,0F4H
00F7 DF
00F8 CC
00F9 D4
00FA DC
00FB E4
00FC EC
00FD F4
00FE FC                 166                 DB      0FCH,0C0H,0C8H,0D0H,098H,0A2H,0CFH,0AAH
00FF C0
0100 C8
0101 D0
0102 98
0103 A2
0104 CF
0105 AA
0106 EB                 167                 DB      0EBH,0E3H,0D8H
0107 E3
0108 D8
                        168
```

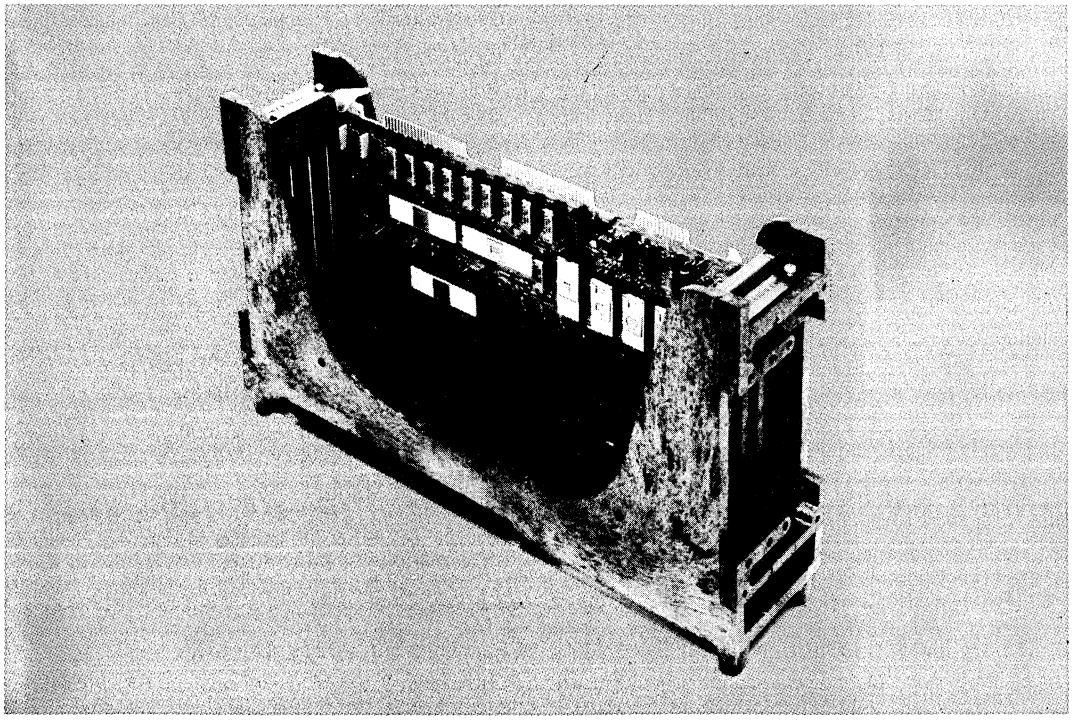
```
                    169 ;*******************************************************************************
                    170 ;                                   TABLE 2
                    171 ;                   ACCEPTABLE OUTPUT CHARACTERS TO DISPLAY
                    172 ;
  0109 C1           173 TABLE2:         DB      0C1H,0C2H,0C3H,0C4H,0C5H,0C6H,0C7H,0C8H
  010A C2
  010B C3
  010C C4
  010D C5
  010E C6
  010F C7
  0110 C8
  0111 C9           174                 DB      0C9H,0CAH,0CBH,0CCH,0CDH,0CEH,0CFH,0D0H
  0112 CA
  0113 CB
  0114 CC
  0115 CD
  0116 CE
  0117 CF
  0118 D0
  0119 D1           175                 DB      0D1H,0D2H,0D3H,0D4H,0D5H,0D6H,0D7H,0D8H
  011A D2
  011B D3
  011C D4
  011D D5
  011E D6
  011F D7
  0120 D8
  0121 D9           176                 DB      0D9H,0DAH,0F1H,0F2H,0F3H,0F4H,0F5H,0F6H
  0122 DA
  0123 F1
  0124 F2
  0125 F3
  0126 F4
  0127 F5
  0128 F6
  0129 F7           177                 DB      0F7H,0F8H,0F9H,0F0H,0FDH,0EBH,0E0H,0EAH
  012A F8
  012B F9
  012C F0
  012D FD
  012E EB
  012F E0
  0130 EA
  0131 EF           178                 DB      0EFH,0EEH,02DH
  0132 EE
  0133 2D
  0000              179                 END     START

PUBLIC SYMBOLS

EXTERNAL SYMBOLS

USER SYMBOLS
BEGIN   A 003B    BUFF    A 3C00    CKFLAG A 005B    CLR     A 00D8    CMDAD   A 00F1    COMPAR A 0073    CONTRO A 00C6
DATAAD A 00F0    DBUFF   A 3D00    DELETE A 00AE    INT     A 00D1    LABEL   A 0064    LOOP1  A 008E    LOOP2  A 00A3
LOOPA  A 009E    LOOPB   A 0086    MATCH  A 0080    MODE0   A 0008    OUTPT   A 0068    PROGCK A 0039    RDFIFO A 0040
RDRAM  A 0060    RDRAMA A 0070    START  A 0000    TABLE1 A 00DE    TABLE2 A 0109    WRRAM  A 0080    WRRAMA A 0090
ZDBUFF A 0055

ASSEMBLY COMPLETE,   NO ERRORS
```

System Packaging and Power Supplies 12

iSBC® 604/614 or (pSBC 604/614*) MODULAR CARDCAGE ASSEMBLIES

- **Interconnects and houses up to four MULTIBUS® boards per cardcage**
- **Connectors allow interconnection of up to four cardcage assemblies for 16 board systems**
- **Strong cardcage structure helps protect installed boards from warping and physical damage**
- **Cardcage mounting holes facilitate interconnection of units**
- **Compatible with 3.5-inch RETMA rack mount increments**
- **Interleaved grounds on backplane minimize noise and crosstalk**
- **Up to 3 CPU boards per system for multiprocessing applications**

The iSBC 604 and iSBC 614 Modular Cardcage Assemblies units provide low-cost, off-the-shelf housing for OEM products using two or more MULTIBUS® boards. Each unit interconnects and houses up to four boards. The base unit, the ISBC 604 Cardcage Assembly, contains a male backplane PC edge connector and bus signal termination circuits, plus power supply connectors. It is suitable for applications requiring a single unit, or may be interconnected with up to three iSBC 614 cardcage assemblies for a four cardcage (16 board) system. The iSBC 614 contains both male and female backplane connectors, and may be interconnected with iSBC 604/614 units. Both units are identical, with the exception of the bus signal terminator feature. A single unit may be packaged in a 3.5 inch RETMA rack enclosure, and two interconnected units may be packaged in a 7 inch enclosure. The units are mountable in any of three planes.



*Same product, manufactured by Intel Puerto Rico, Inc.

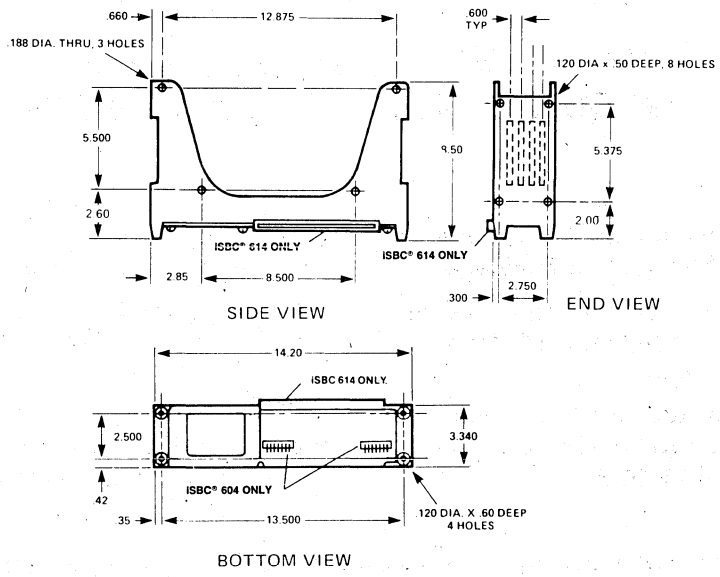


Figure 1. iSBC® 604/814 Cardcage Assembly Dimensions

SPECIFICATIONS

Backplane

Bus Lines — All MULTIBUS system bus address, data, and command bus lines are bussed to all four connectors on the printed circuit backplane

Power Connectors — G for ground, +5V, −5V, + 12V, −12V, and −10V power supply lines

iSBC 604 — Bus signal terminators, backplane male PC edge connector only, and power supply headers

iSBC 614 — Backplane male and female connectors and power supply headers

Mating Power Connectors

AMP	Connector	87159-7
	Pin	87023-1
	Polarizing key	87116-2
Molex	Connector	09-50-7071
	Pin	08-50-0106
	Polarizing key	15-04-0219

Note
1. Pins from a given vendor may only be used with connectors from the same vendor.

Bus Arbitration — Serial; up to 3 CPU masters

Equipment Supplied

iSBC 604 or iSBC 614 Cardcage
Schematic

Physical Dimensions

Height — 8.5 in. (21.59 cm)
Width — 14.2 in. (36.07 cm)
Depth — 3.34 in. (8.48 cm)
Weight — 35 oz. (992.23 gm)
Card Slot Spacing — 0.6 in.

Environmental Characteristics

Operating Temperature — 0°C to 55°C

Reference Manual

9800708 — iSBC 604/614 Cardcage Hardware Reference Manual (ORDER SEPARATELY)

ORDERING INFORMATION

Part number	Description
SBC 604	Modular Cardcage Assembly (Base Unit)

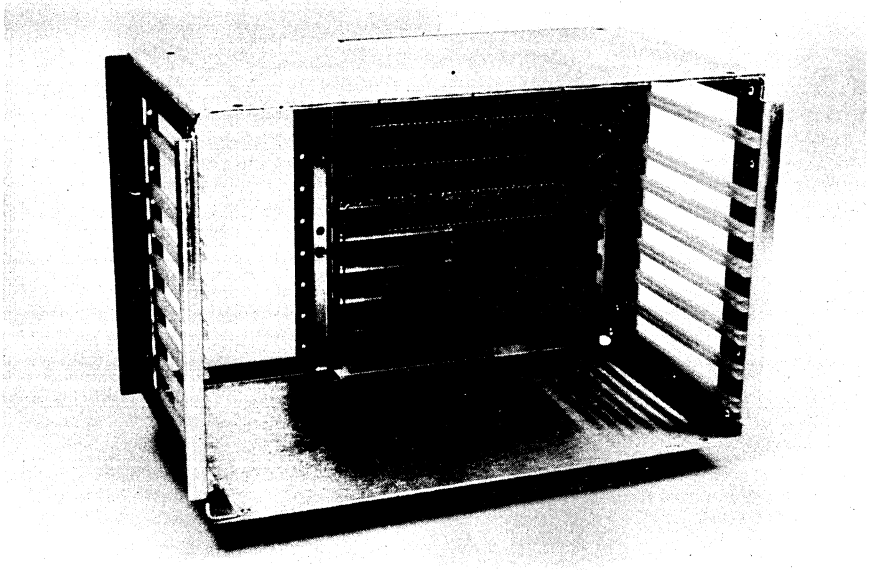
Part Number	Description
SBC 614	Modular Cardcage Assembly (Expansion Unit)

iSBC® 608/618 CARDCAGES

- **Houses eight MULTIBUS® iSBC® boards in an aluminum package**
- **Board-to-board clearance for iSBC® MULTIMODULE™ boards on all slots**
- **Board-to-board clearance for iSBX™ MULTIMODULE™ boards on two slots**
- **Parallel priority circuitry for up to eight Multimaster iSBC®boards**
- **Enhanced bus noise immunity for high speed systems**
- **Plug on iSBC® 618 unit for up to sixteen board systems**
- **NEMA-type backwall or 19-inch rack mount hardware included**
- **Signal line termination circuitry on iSBC® 608 Cardcage**

Intel's iSBC 608/618 Cardcages are matched to the latest generation of iSBC/iSBX boards which mount in the MULTIBUS system bus. These products provide several features which make them the industry's leading price/performance cardcage product. MULTIMODULE board clearance, parallel priority circuitry, enhanced backplane noise immunity, and precision fit card guides are a few of the distinctions which make this the industry's better product.

The iSBC 608 Cardcage is the base unit, housing up to eight iSBC boards and their MULTIMODULE boards. Additionally, this base unit includes mounting hardware and fan mounting bracketry. The iSBC 618 is the expansion unit, providing eight additional iSBC board slots to the iSBC 608 Cardcage for a total of sixteen board slots which can be NEMA-type backwall or 19-inch rack mounted. This is accomplished with the mounting hardware of the iSBC 608 Cardcage. The iSBC 618 expansion unit also includes fan mounting bracketry.



March, 1982
Order Number: 210373-001

FUNCTIONAL DESCRIPTION

Mechanical Aspects

The iSBC 608/618 Cardcages provide housing and a MULTIBUS system bus for up to sixteen single board computers and their MULTIMODULE boards. The iSBC 608 unit and iSBC 618 unit offer board-to-board clearance (0.8 inches or greater) on all eight slots for iSBC MULTIMODULE boards. Two slots provide clearance (1.2 inches or greater) for iSBX MULTIMODULE boards as shown in Figure 1. Each cardcage includes precision fitted nylon cardguides for secure board fit and accurate MULTIBUS board pin alignment. Fan mounting bracketry is also included with each cardcage. This bracketry allows the mounting of several industry standard fans. The iSBC 608 Cardcage base unit includes aluminum mounting hardware for NEMA-type backwall mounting, or anchoring a sixteen slot iSBC 608/618 combination in a standard 19-inch rack.

Electrical Aspects

The iSBC 608/618 Cardcages implement a parallel priority resolution scheme by using plug-in jumper connections. There are six different priority schemes allowed, each requiring a different jumper configuration. In systems where an iSBC 618 Cardcage is attached to the base unit, the base unit will have lower priority overall. That is, master boards in the iSBC 608 base unit may gain control of the MULTIBUS lines only when no boards in the iSBC 618 expansion unit are asserting the bus request (BREQ/) signal.

Noise-minimizing ground traces are strategically interleaved between signal and address lines on these backplanes. This provides the enhanced noise immunity and minimized signal-to-signal coupling which is important in high speed, high board count microcomputer systems.

The iSBC 608/618 Cardcages provide power connector lug bolts for +5 VDC and ground. The lug bolts, compared to other power connection methods, help transfer higher amounts of current. Other voltages (±12 VDC, −5 VDC) are connected via a mating power connector plug as shown in Figure 2.

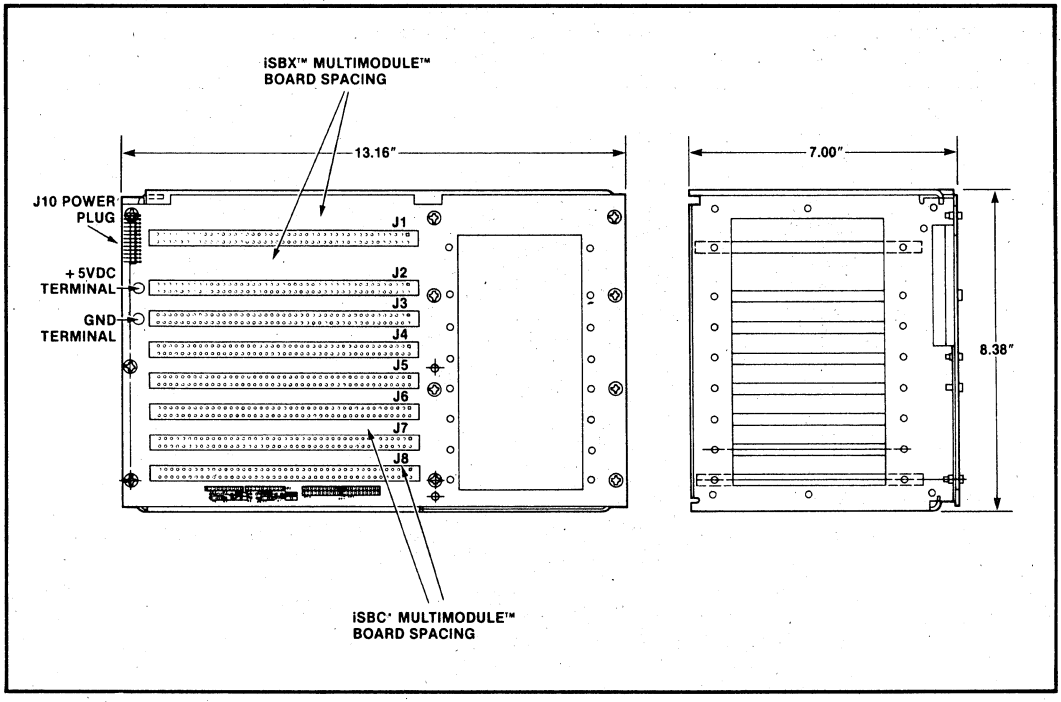


Figure 1. iSBC® 608/618 Cardcages Dimensions

SPECIFICATIONS

Bus Lines

All MULTIBUS (IEEE 796) system bus address and command lines are bussed to each of the eight MULTIBUS connectors on the backplane. Ground traces are interleaved among these signal lines and bussed to the backplane edge connector for interconnection of the iSBC 608 and iSBC 618 backplane.

Power Connectors

Ground (0V), +5V, −5V, +12V, −12V power supply header stakes and power lug bolts are provided on the iSBC 608/618 Cardcages as shown in Figure 2.

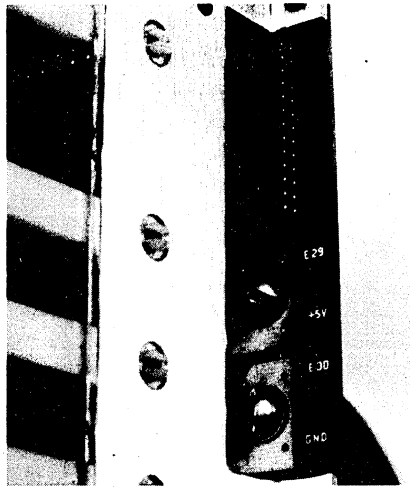


Figure 2. Power Header Stakes and Lugs

Environmental Characteristics

Operating Temperature — 0°C to 55°C

Storage Temperature — −40°C to 85°C

Humidity — 50% to 95% non-condensing at 25°C to 40°C

Vibration and Shock — 2G max. through 50 Hz

Physical Characteristics

SLOT-TO-SLOT DIMENSIONS (See Figure 1)

Top-J1 — 1.200 in (to center)

J1-J2 — 1.300 in (center to center)

J8-Bottom — 0.700 in (to center)

All Others — 0.800 (center to center)

PHYSICAL DIMENSIONS

Height — 8.38 in (21.29 cm)

Length — 13.16 in (33.43 cm)

Width — 7.50 in (19.05 cm)

Weight — 3.50 lbs (1.59 kg)

Shipping Weight — 5.75 lbs (2.61 kg)

Equipment Supplied

iSBC® 608 BASE UNIT

Eight-Slots — Two at greater than 1.2 inches; six at 0.8 inches

Male Backplane Connector — For expansion with iSBC 618 cardcage

Parallel Priority Circuitry — Eight slots are configurable via the use of jumper stakes. Six priority schemes allowed

Construction Materials —

Aluminum card housing
Nylon card guides
Power connector header stakes and lug bolts

Accessories

iSBC® 618 EXPANSION UNIT

Eight-Slots — Two at greater than 1.2 inches; six at 0.8 inches

Female Backplane Connector — For expansion to iSBC 608 base unit

Parallel Priority Circuitry — Eight slots are configurable via the use of jumper stakes. Six priority schemes allowed

Construction Materials —
Aluminum card housing
Nylon card guides
Power connector header stakes and lug bolts
Fan Mounting Hardware
Schematic

User-Supplied Equipment

MATING POWER CONNECTORS

Vendor	Part Number
3M	3399-6026
Ansley	609-2600M
Berg	65485-009

MOUNTABLE FANS

Vendor	Part Number
Rotron	SU2A1-028267
Torin	TA300-A30473-10
Pamotor	8506D

Reference Manual

144261-001 — iSBC 608/618 Cardcages Hardware Reference Manual (order separately)

Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051

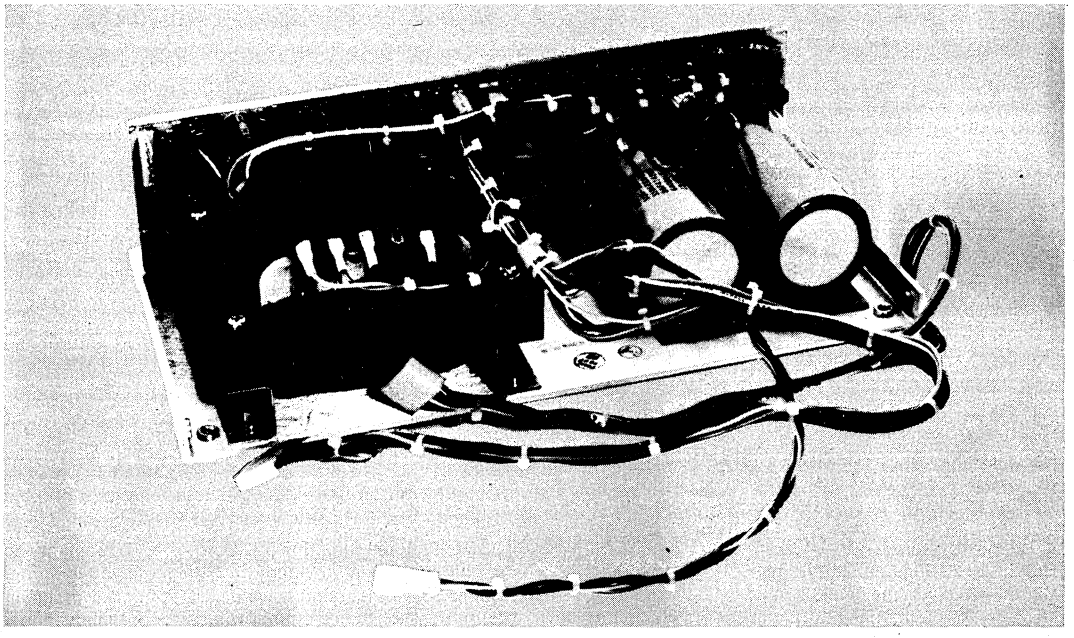
ORDERING INFORMATION

Part Number	Description
SBC 608	Cardcage (base unit)
SBC 618	Cardcage (expansion unit for iSBC 608)

iSBC® 635
POWER SUPPLY

- **Compact single chassis**
- **±5V and ±12V output voltages**
- **Sufficient power for one fully loaded Intel single board computer plus residual power for up to three Intel iSBC® expansion boards**
- **Current limiting and overvoltage protection on all outputs**
- **UL Listed and CSA Certified**
- **DC power cables and connectors mate directly to iSBC® 604 Modular Cardcage/ Backplane assembly**
- **"AC low" power failure TTL logic level output provided for system power-down control**
- **100V, 115V, 215V, and 230V AC operation**
- **50 Hz or 60 Hz input**

The iSBC 635 Power Supply provides low cost, off-the-shelf, single chassis power generation for OEM products using Intel single board computers. The iSBC 635 supply provides regulated DC output power at +12V, +5V, −12V, and −12V levels. The current capabilities of each of these output levels have been chosen to provide power over a 0°C to +55°C temperature range for one Intel single board computer fully loaded with I/O line terminators and drivers and EPROMs, plus residual capability for most combinations of up to three iSBC memory, I/O or combination expansion boards. Current limiting and overvoltage protection is provided on all outputs. Access for AC input is provided via a standard 4-pin keyed connector. DC output power levels are provided on cables with keyed connectors directly compatible with the iSBC 604/614 Modular Cardcage/Backplane assembly. The iSBC 635 supply includes logic whose purpose is to sense system AC power failure and generate a TTL signal for clean system power-down control.



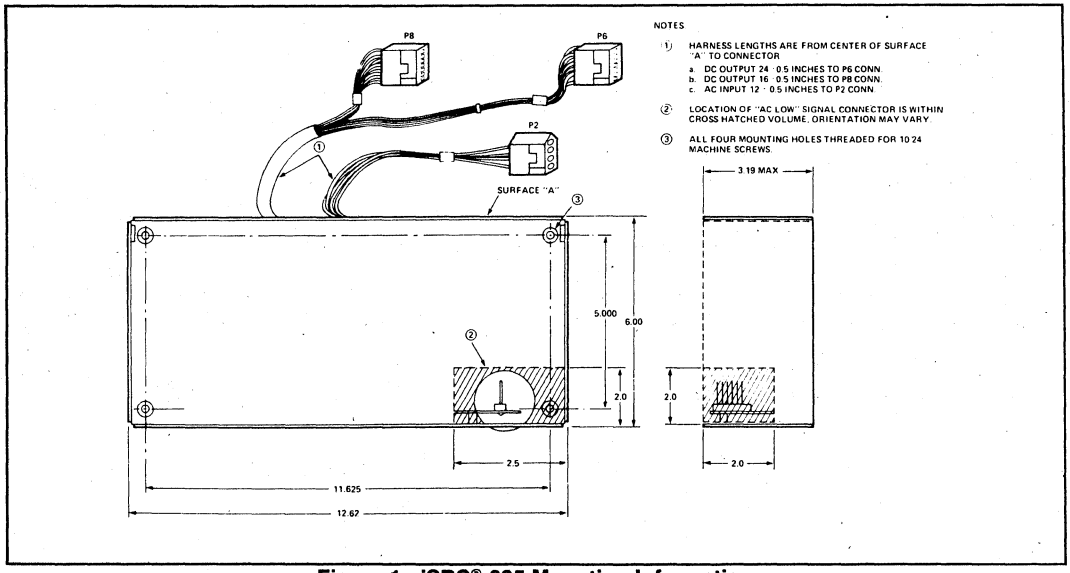


Figure 1. iSBC® 635 Mounting Information

SPECIFICATIONS

Mating Connectors[1]

AC Input

Connector	Molex	03-09-1042 or equivalent
Pin	Molex	02-09-1118 or equivalent (18 to 22 gauge wire)

DC Output[2]

Header	Molex	09-66-1071
	AMP	87194-6

"AC Low" Control

Connector	Molex	09-50-7071
	AMP	87159-7
Polarizing key	Molex	15-04-0219
	AMP	87116-2
Pin	Molex	08-50-0106 (18 to 22 gauge wire)
	AMP	87023-1 (18 to 22 gauge wire)

Notes

1. Pins from a given vendor may only be used with connectors from the same vendor.

2. iSBC 635 DC output connectors are directly compatible with power input power connectors on iSBC 604 Modular Cardcage/Backplane assembly. Two connectors are provided.

Physical Characteristics

Height — 3.19 in. max (8.11 cm)
Width — 6.03 in. max (15.32 cm)
Depth — 12.65 in. max (32.12 cm)
Weight — 13 lb (5.90 kgm)

Electrical Characteristics

Input Power — Frequency: 47-63 Hz. Voltage (Nominal) (Single Phase): 100, 115, 215, or 230 VAC ±10%

Output Power:

Nominal Voltage	Current (AMPS)(MAX)	Current Limit Range (AMPS)	Max Short Circuit (AMPS)	Over-Voltage Protection
+12	2.0	2.1-3.0	1.0 (Foldback)	+14 to +16 V
+ 5	14.0	14.7-21.0	7.0 (Foldback)	+5.8 to +6.6 V
- 5	0.9	0.9-1.4	1.4	-5.8 to -6.6 V
-12	0.8	0.8-1.2	1.2	-14 to -16V

Combined Line/Load Regulation — ±1% at ±10% static line change and ±50% static load change, measured at the output connector (±0.2% measured at the power supply under the same conditions).

Remote Sensing — Provided for +5VDC output line regulation.

Output Ripple and Noise — 10 mV peak-to-peak maximum (DC to 500 KHz)

Output Transient Response — Less than 50 μsec for ±50% load change

Output Transient Deviation — Less than ±5% of initial voltage for ±50% load change.

Power Failure Indication (AC Low) — A TTL open collector high signal is provided when the input voltage drops below 90% of its nominal value. DC voltages will remain within 5% of their nominal values for 3.0 milliseconds (minimum) after AC low goes true.

The "AC Low" signal will reset to a TTL low level when the AC input voltage is restored and after all output voltages are within specified regulation.

The "AC Low" threshold is adjustable for optimum powerdown performance at other input combinations (i.e. 100 VAC, 215 VAC, 50 Hz).

Environmental Characteristics

Operating Temperature — 0°C to +55°C with 35 CFM moving air

Non-Operating — −40°C to +85°C

Equipment Supplied

iSBC 635 Power Supply with AC and DC cables and connectors attached as shown in Figure 1.

Reference Manual

9800298C - iSBC 635 Power Supply Hardware Reference Manual (Order Separately)

Manuals may be ordered from any Intel sales representative distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

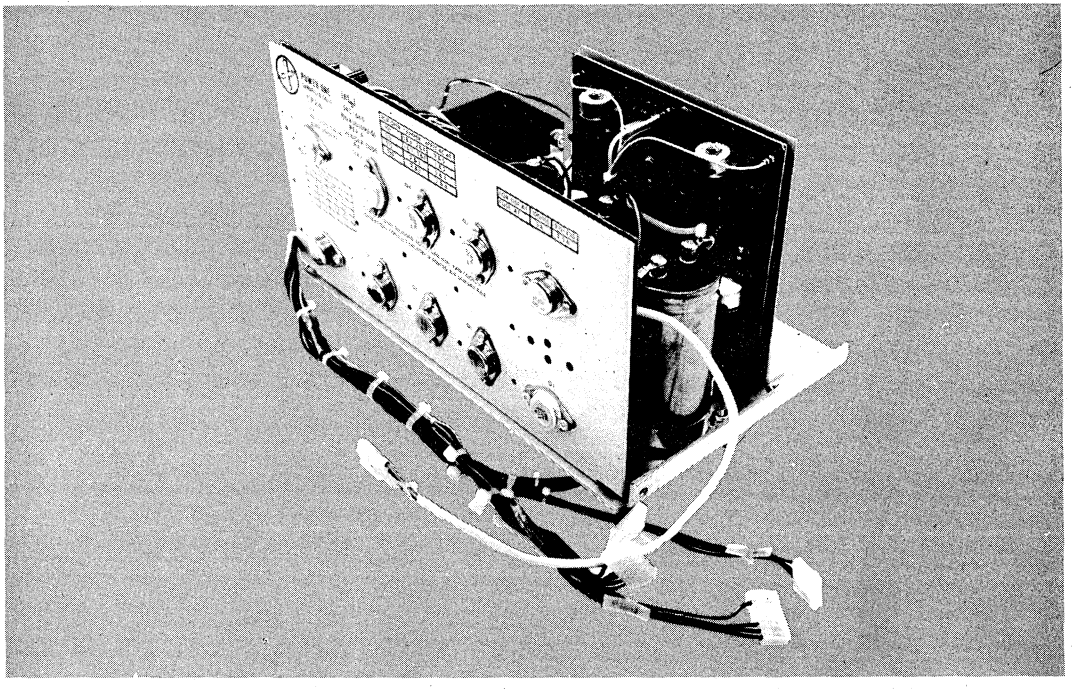
ORDERING INFORMATION

Part Number	Description
SBC 635	Power Supply

iSBC® 640
POWER SUPPLY

- ± 5V and ± 12V output voltage
- Sufficient power for 8-12 MULTIBUS® computer, memory, and peripheral boards
- Current limiting and overvoltage protection on all outputs
- UL Listed and CSA Certified
- "AC low" power failure TTL logic level output provided for system power-down control
- DC power cables and connectors mate directly to iSBC 604/614 and iSBC 608/618 Modular Cardcage/Backplane assemblies
- 100, 120, 220, and 240V AC operation,
- 50 Hz or 60 Hz input

The iSBC 640 Power Supply provides low cost, off-the-shelf, single chassis power generation for OEM and industrial system products using Intel single board computers. The iSBC 640 supply provides regulated DC output power at +12V, +5V, and −5V and −12V levels. The current capabilities of each of these output levels has been chosen to provide power over a 0°C to +55°C temperature range for one fully loaded Intel single board computer, plus residual capability for most combinations of up to eleven iSBC memory, I/O, or combination expansion boards. Current limiting and overvoltage protection is provided on all outputs. Access for AC input is provided via a standard 4-pin keyed connector. DC output power levels are provided on cables with keyed connectors directly compatible with the iSBC 604/614 and iSBC 608/618 Modular Backplane/Cardcage assemblies. The iSBC 640 supply includes logic whose purpose is to sense system AC power failure and generate a TTL signal for clean system power-down control.



SPECIFICATIONS

Electrical Characteristics

Input Power
Frequency: 50 Hz ± 5%, 60 Hz ± 5%
Voltage: 100/120/220/240 VAC ± 10%
Via user configured wiring options

Output Power

Nominal Voltage	Current (Amps)(Max)	Current Limit Range (Amps)	Short Circuit (Amps)(Max)	Overvoltage Protection
+ 12V	4.5A	4.7- 6.8	2.3	15V ± 1V
+ 5V	30A	31.5-45.0	15.0	6.2V ± 0.4V
- 5V	1.75A	1.8- 3.2	0.9	- 6.2V ± 0.4V
- 12V	1.75A	1.8- 3.2	0.9	- 15V ± 1V

Combined Line/Load Regulation — ±1% at ±10% static line change and ±50% static load change, measured at the output connector (±0.2% measured at the power supply under the same conditions).

Remote Sensing — Provided for +5 VDC output line regulation.

Output Ripple and Noise — 10 mV peak-to-peak maximum (DC to 500 KHz)

Output Transient Response — Less than 50 μsec for ±50% load change.

Output Transient Deviation — Less than ± 10% of initial voltage for ± 50% load change.

Power Failure Indication (AC Low) — A TTL open collector high signal is provided when the input voltage drops below 90% of its nominal value. DC voltages will remain within 5% of their nominal values for 3.0 milliseconds (minimum, 7.5 ms typical) after AC Low goes true.

The "AC Low" signal will reset to a TTL low level when the AC input voltage is restored and after all output voltages are within specified regulation.

The "AC Low" threshold is adjustable for optimum powerdown performance at other input combinations (i.e. 100 VAC, 220 VAC, 50 Hz).

Mating Connectors[1]

AC Input

Housing	Molex	03-09-2042 or equivalent
Pin	Molex	02-09-2118 or equivalent (18 to 22 gauge wire)

DC Output[2]

Housing	Molex	26-03-3071
	Amp	3-87025-3
Pins	Molex	08-50-0187 or 08-50-0189
	Amp	87023-1
Key	Molex	15-04-9209
	Amp	87116-2

Compatible with Molex 09-66-1071 Header

Notes
1. Pins from given vendor may only be used with connectors from the same vendor.
2. iSBC 640 DC output connectors are directly compatible with input power connectors on iSBC 604/614 and iSBC 608/618 Modular Card-cage/Backplane assemblies. Four connectors are provided.

Physical Characteristics

Height — 6.66 in. max. (16.92 cm)
Width — 8.19 in. max. (20.80 cm)
Depth — 12.65 in. max. (32.12 cm)
Weight — 30 lbs. max (13.63 kg)

Environmental Characteristics

Temperature — 0°C to 55°C with 55 CFM moving air
Non-Operating — - 40°C to + 85°C

Equipment Supplied

iSBC 640 Power Supply with AC and DC cables with keyed connectors.

Reference Manuals

9800803 — iSBC 640 Power Supply Hardware Reference Manual (order separately)

9800798 — iCS 80 Systems Site Planning and Installation Manual (for installation of iSBC 640 supply into iCS 80 Industrial Chassis) (Order Separately)

Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

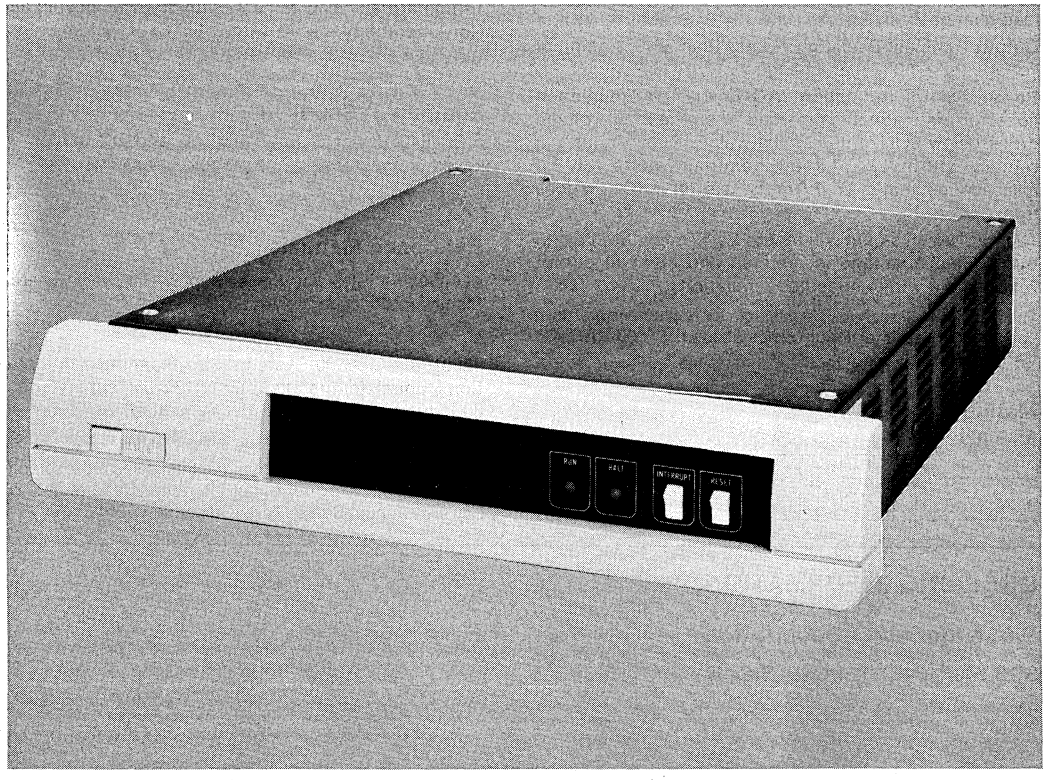
ORDERING INFORMATION

Part Number	Description
SBC 640	Power Supply

iSBC® 655
SYSTEM CHASSIS

- **A rack-mountable package for Intel microcomputer system**
- **Provides the Intel MULTIBUS® structure used on the single board computers**
- **Compact single chassis power supply with all standard iSBC® board voltages**
- **Attractive front panel with control switches and indicator lights**
- **100, 120, 220, and 240V A.C.**
- **19-inch rack mountable**
- **Forced-air cooling**

The iSBC® 665 System Chassis is an attractive 3.5″ high unit designed for use in Intel Microcomputer Systems. The Chassis' four slots accommodate both single board computers and expansion boards which provide additional I/O, memory, or peripheral controller functions. The iSBC 655 System Chassis will accept all Intel boards using the MULTIBUS® architecture. DC power is provided at ±5VDC and ±12VDC levels, at current levels commensurate with typical combinations of four boards. The chassis is designed to provide adequate cooling to both power supply and circuit boards over external temperatures ranging from 0°C to 50°C. Current limiting and over-voltage protection are provided on all outputs. The power supply recognizes an AC power failure condition and provides a TTL signal sufficiently in advance of DC power failure to allow orderly system shut-down. For user convenience, system RESET and INTERRUPT switches are provided on the front panel to facilitate system restarts and provide for operator intervention. RUN and HALT LED indicators are driven to indicate the operational status of the single board computer.



SEPTEMBER, 1984
ORDER NUMBER 280068-001

SPECIFICATIONS

Electrical

Input Power — Frequency: 47-63 Hz. Voltage (Nominal) (Single Phase): 100, 115, 215, or 230 VAC ± 10%

Output Power:

Nominal Voltage	Current (AMPS) (MAX)	Current Limit Range (AMPS)	Max Short Circuit (AMPS)	Over-Voltage Protection
+12	2.0	2.1-3.0	1.0 (Foldback)	+14 to +16 V
+ 5	14.0	14.7-21.0	7.0 (Foldback)	+5.8 to +6.6 V
+5	0.9	0.9-1.4	1.4	−5.8 to −6.6 V
+12	0.8	0.8-1.2	1.2	−14 to −16 V

Combined Line/Load Regulation — ± 1% at ± 10% static line change and ± 50% static load change, measured at the output connector (± 0.2% measured at the power supply under the same conditions).

Remote Sensing — Provided for +5VDC output line regulation.

Output Ripple and Noise — 10 mV peak-to-peak maximum (DC to 500 KHz)

Output Transient Response — Less than 50 μsec for ± 50% load change

Output Transient Deviation — Less than ± 5% of initial voltage for ± 50% load change

Power Failure Indication (AC Low) — ATTL open collector high signal is provided when the input voltage drops below 90% of its nominal value. DC voltages will remain within 5% of their nominal values for 3.0 milliseconds (minimum, 7.5 ms typical) after AC LOW goes true.

Mechanical

Height — 3.5 inches (8.9 cm)

Width — 19 inches (48.3 cm) at Front Panel, 17 inches (43.2 cm) behind Front Panel

Depth — 20 inches (50.8 cm) with all protrusions

Weight — 37 pounds (17 Kg)

Cardcage — 4 board capacity at 0.6 in spacing

Environmental

Temperature — Operating: 0°C to 50°C.
Non-Operating: 40°C to 85°C

Relative Humidity — Up to 90%, non-condensing

Equipment Supplied

iSBC 655 System Chassis with power supply, cardcage/backplane, dual fans, pop-off front panel and top cover

Reference Manual (Not Supplied)

9800709A — iSBC 655 System Chassis Hardware Reference Manual

ORDERING INFORMATION

Part Number	Description
SBC 655	iSBC 655 System Chassis

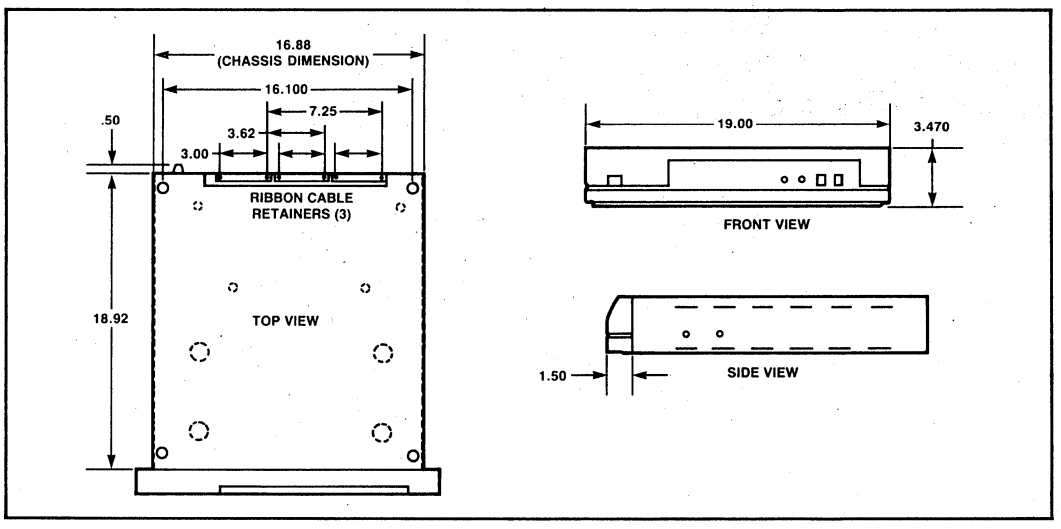
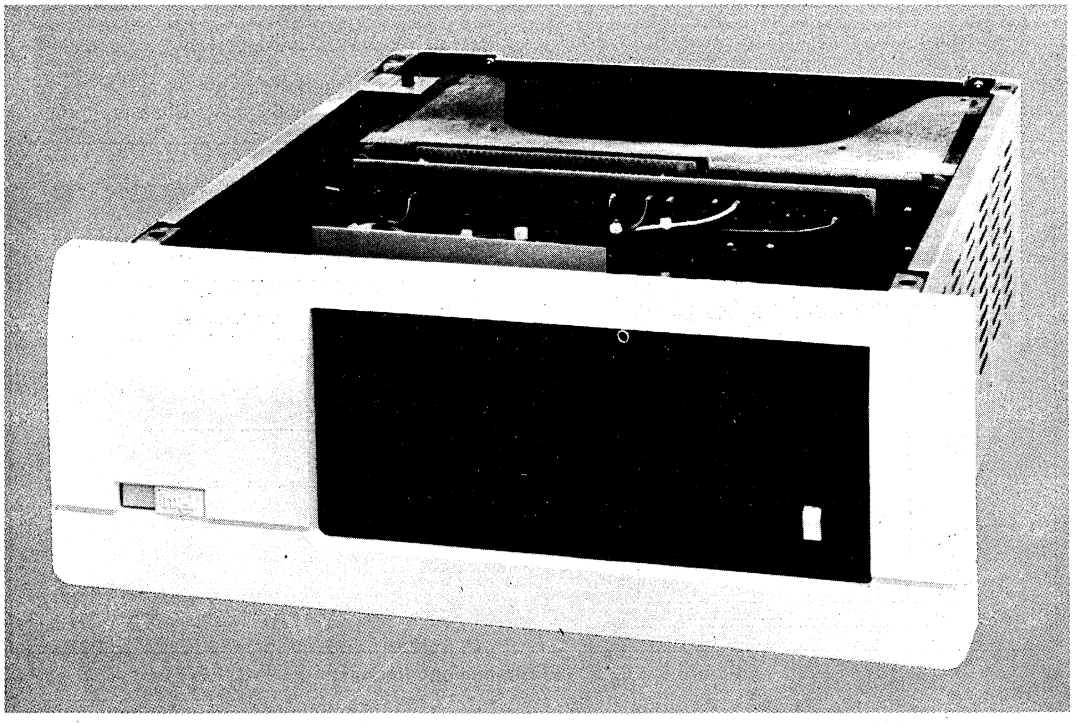


Figure 1. iSBC® 655 Dimensions (inches)

iSBC® 660
SYSTEM CHASSIS

- **Eight-slot cardcage and backplane for iSBC® computers and expansion boards**
- **Heavy duty power supply with all standard iSBC® voltages**
- **Compatible with all Intel single board computers**
- **Forced-air cooling**
- **Attractive, versatile pop-off front panel**
- **19-inch wide rack mountable chassis**
- **Horizontal board mounting for compactness**
- **100/120/220/240 VAC, 50/60 Hz operation**

The iSBC 660 System Chassis is an attractive, 7-inch high system chassis designed for use with Intel OEM computers. It has eight slots for single board computers, memory, I/O, or other expansion modules. The iSBC 660 is ideal for applications requiring multiple board solutions. DC power output is provided at +12V, +5V, −12V, and −5V levels. The current capabilities of each of these output levels have been chosen to provide power over a 0°C to 50°C temperature range for the majority of applications requiring combinations of computers, memories, peripherals, and other I/O capabilities. Current limiting and over-voltage protection is provided at all outputs. Standard logic recognizes a system AC power failure and generates a TTL signal for use in power-down control. For user convenience, a reset switch is provided on the front panel. The reset signal generated and sent to the system bus can be used for external system control.



AFN-00285A

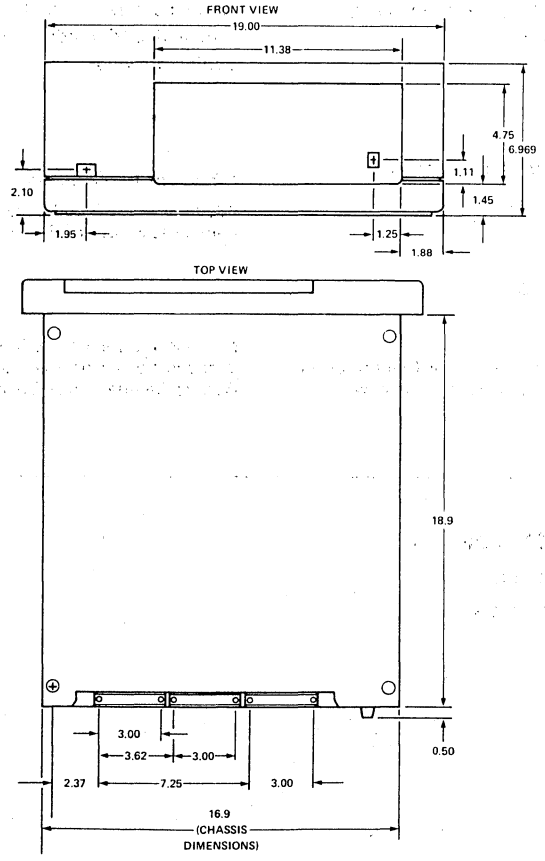


Figure 1. iSBC® System Chassis Dimensions

SPECIFICATIONS

Electrical Characteristics

Input Power

Frequency: 50 Hz ± 5%, 60 Hz ± 5%

Voltage: 100/120/220/240 VAC ± 10% via user configured wiring options

Output Power

Power	Output Current (Max)	Current Limit (Amps)	Over-Voltage Protection
+ 12V	4.5A	5.4	15V ± 1V
+ 5V	30A	3.6	6.2V ± 0.4V
- 5V	1.75A	2.1	-6.2V ± 0.4V
- 12V	1.75A	2.1	-15V ± 1V

Combined Line/Load Regulation — ± 1% at ± 10% static line change and ± 50% static load change, measured at the output connector (± 0.2% measured at the power supply under the same conditions).

Remote Sensing — Provided for + 5 VDC output line regulation.

Output Ripple and Noise — 10 mV peak-to-peak maximum (DC to 500 kHz).

Output Transient Response — Less than 50 μs for ± 50% load change.

Output Transient Deviation — Less than ± 5% of linitial voltage for ± 50% load change.

Power Failure Indication (AC Low) — A TTL open collector high signal is provided when the input voltage drops below 90% of its nominal value. DC voltages will remain within 5% of their nominal values for 3.0 milliseconds (minimum) after AC low goes true.
The "AC Low" signal will reset to a TTL low level when the AC input voltage is restored and after all output voltages are within specified regulation.

The "AC Low" threshold is adjustable for optimum power-down performance at other input combinations (i.e. 100 VAC, 220 VAC, 50 Hz).

Humidity — Up to 90% relative, non-condensing

Physical Characteristics

Height — 7 in. (17.8 cm)

Width
At Front Panel: 19 in. (48.3 cm)
Behind Front Panel: 17 in. (43.2 cm)

Depth — 20 in. (50.8 cm) with all protrusions

Environmental Characteristics

Temperature
Operating: 0°C to 50°C
Non-Operating: −40°C to +85°C

Equipment Supplied

iSBC 660 System Chassis with power supply, 8 slot cardcage assembly, dual fans, pop-off front panel and top cover

I/O connectors for single board computers
Schematics for cardcage/backplane, chassis
Outline drawing

Reference Manuals

9800505 — iSBC 660 System Chassis Hardware Reference Manual (ORDER SEPARATELY)

Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

ORDERING INFORMATION

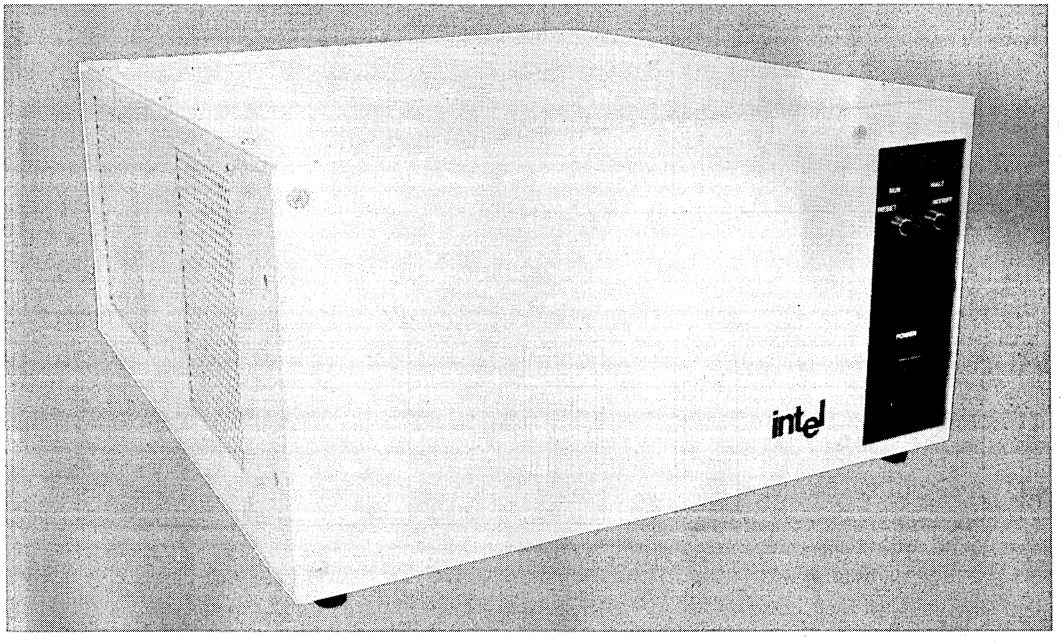
Part Number	Description
SBC 660	System Chassis

iSBC® 661
SYSTEM CHASSIS

- **Eight-slot MULTIBUS® chassis with parallel priority circuitry**
- **UL, FCC and CSA approved for data processing equipment**
- **230 watt power supply with power fail warning**
- **Designed for slide rack mounting or table-top use**
- **Extra-wide cardcage slot spacing for iSBX™ MULTIMODULE™ board clearance**
- **Configurable for front or rear access to MULTIBUS® circuit boards**
- **Five connector ports for I/O cabling**
- **Operational from 47 Hz to 63 Hz, 100/120/220/240 VAC ± 10%**

The iSBC® 661 System Chassis is an advanced MULTIBUS® (IEEE) 796 chassis which incorporates unique usability and service features not found on competitive products. This chassis is designed for rack-mount or table-top applications and reliably operates up to an ambient temperature of 50°C. Additionally, this system chassis is certified by UL, CSA, and FCC for data processing equipment.

An application requiring multiprocessing will find this eight-slot MULTIBUS chassis particularly well suited to its needs. Parallel priority bus arbitration circuitry has been integrated into the backplane. This permits a bus master to reside in each slot. Extra-wide inter-slot spacing on the cardcage allows the use of plug-on MULTIMODULE™ boards without blocking adjacent slots. For this reason, the iSBC 661 System Chassis provides the slot-functionality of most 16-slot chassis. Standard logic recognizes a system AC power failure and generates a TTL signal for use in power-down control. Additionally, current limiting and over-voltage protection are provided at all outputs.



OCTOBER 1984
ORDER NUMBER: 210866-002

FUNCTIONAL DESCRIPTION

Mechanical Features

The iSBC 661 System Chassis houses, cools, powers, and interconnects up to eight iSBC single board computers and their MULTIMODULE boards for the MULTIBUS System Bus. Based on Intel's iSBC 608 Cardcage, the chassis provides 0.8 inches of board center-to-center clearance on six slots, and 1.2 inches or more of center-to-center clearance on two slots. This permits the users of standard MULTIMODULE boards and custom wire-wrap boards to plug into the MULTIBUS System Bus without blocking adjacent slots. All slots provide enough clearance for iSBC MULTIMODULE boards, and two slots can accommodate iSBX MULTIMODULE boards.

High-technology MULTIBUS applications requiring rack-mount, or laboratory table-top use will find the iSBC 661 System Chassis ideal. Standard 19″ slide-rack mounting is possible with user-provided slides attached to the side panels. Slide mounting holes are provided in the chassis for the slide-rails listed under User Supplied Options. Rubber feet are included on the chassis for convenient table-top use.

The chassis is constructed of burnished aluminum which has been coated with corrosion-resistant chromate. It contains a system control module which presents the front panel control switches to the user, and holds the I/O cabling bulkhead to the rear. The chassis has the unique feature of being configurable for either front or rear access to MULTIBUS circuit boards.

This is accomplished by a simple procedure involving removal of the system control module, reversing it end-for-end, and re-securing it to the chassis. The system chassis is shipped in a configuration such that the MULTIBUS boards are installed from the front.

Electrical Features

The iSBC 661 System Chassis is powered by the iSBC 640 power supply. This is a standard Intel power supply which has been adopted by several MULTIBUS vendors throughout the industry. It supplies 230 watts of

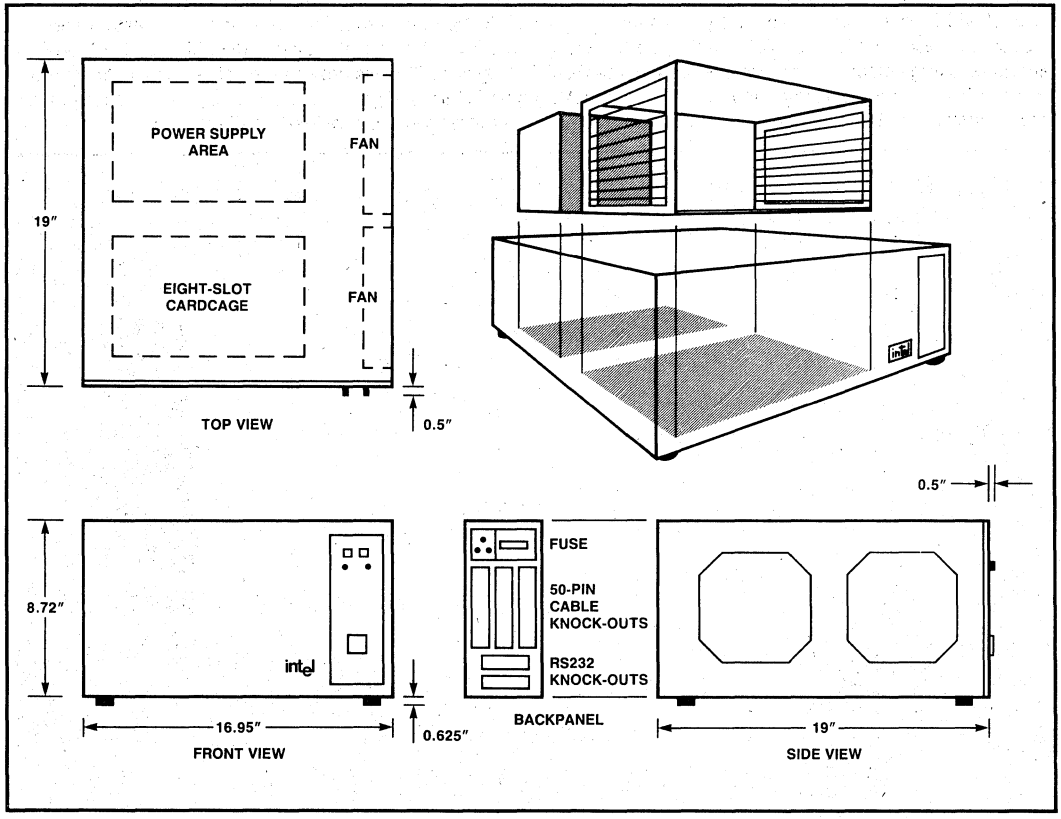


Figure 1. iSBC® 661 System Chassis Dimensions

power, power fail warning, and remote sensing of +5 volts. Its electrical and operational parameters are listed under Specifications.

The cardcage of the iSBC 661 System Chassis implements a user-changeable parallel priority bus arbitration scheme by using plug-in jumper connections. Six different priority schemes are allowed, each scheme fixing the priority of the eight MULTIBUS board slots. Bus contention among eight bus-masters in a multiprocessing environment can be be managed using this approach.

Noise minimizing ground traces are strategically interleaved between signal and address lines on the system bus. This provides the enhanced noise immunity and minimized signal-to-signal coupling which is particularly important in high speed, high board count microcomputer systems.

SPECIFICATIONS

Electrical Parameters

OUTPUT POWER

Table 1. Output Power Levels iSBC® 661-1

Voltage	Output Current (max.)	Current Limits (amps)	Over-Voltage Protection
+12V	4.5A	4.7-6.8	15V ± 1V
+5V	30.0A	31.5-45.0	6.2V ± 0.4V
-5V	1.75A	1.8-3.2	-6.2V ± 0.4V
-12V	1.75A	1.8-3.2	-15V ± 1V

OPERATIONAL PARAMETERS

Input AC Voltage — 100/120/220/240 VAC ± 10% (User selects via external switch), 47-63 Hz

Power-Fail Indication and Hold-Up Time (triggered at 90% of VAC in) — TTL O.C. High 3 msec. (min.)

Output Ripple and Noise — 1% Peak-to-Peak output nominal (DC to 0.5 MHz)

Operational Temperature — 0°C to 50°C

Storage Temperature — -40°C to 70°C

Operational Humidity — 10% to 85% relative, non-condensing

Remote Sensing — Provided for +5 VDC

Output Transient Response — 50 μsec or less for ±50% load change

PHYSICAL CHARACTERISTICS

Width — 16.95 inches (43.05 cm)

Height — 8.72 inches (22.2 cm)

Depth — 19.00 inches (48.3 cm)

Weight — 41 pounds (21 kg)

Shipping Weight (approx.) — 50 pounds (25 kg)

Equipment Supplied

iSBC® 661-1 — Eight-slot MULTIBUS system chassis chassis with parallel priority arbitration circuitry and 230 watt linear power supply

REFERENCE MANUAL (Not included: order separately)

145340-001 — iSBC 661 System Chassis Hardware Reference Manual

Reference manual may be ordered from any Intel sales representative, distributor office or from Intel Literature Department.

In North America: Intel Corp. Literature Department
3065 Bowers Ave.
Santa Clara, California 95051
Phone: (408) 987-8080

In Europe: Intel Corp. S.A. Literature Department
Rue du Moulin A Papier 51
Boite 1
B-1160 Brussels, Belgium
Phone: 322-661-07-11

In the Orient: Intel Corp. Literature Department
5-6 Tokodai, Toyosato-cho
Tsukuba-gun, Ibaragi-ken 300-26
Japan
Phone: 81-29747-8591

User Supplied Options

Compatible Rack-Mount Slides — Chassis Trak, Inc., P.O. Box 39100, Indianapolis, IN 46239; Part No. C 300 S 122

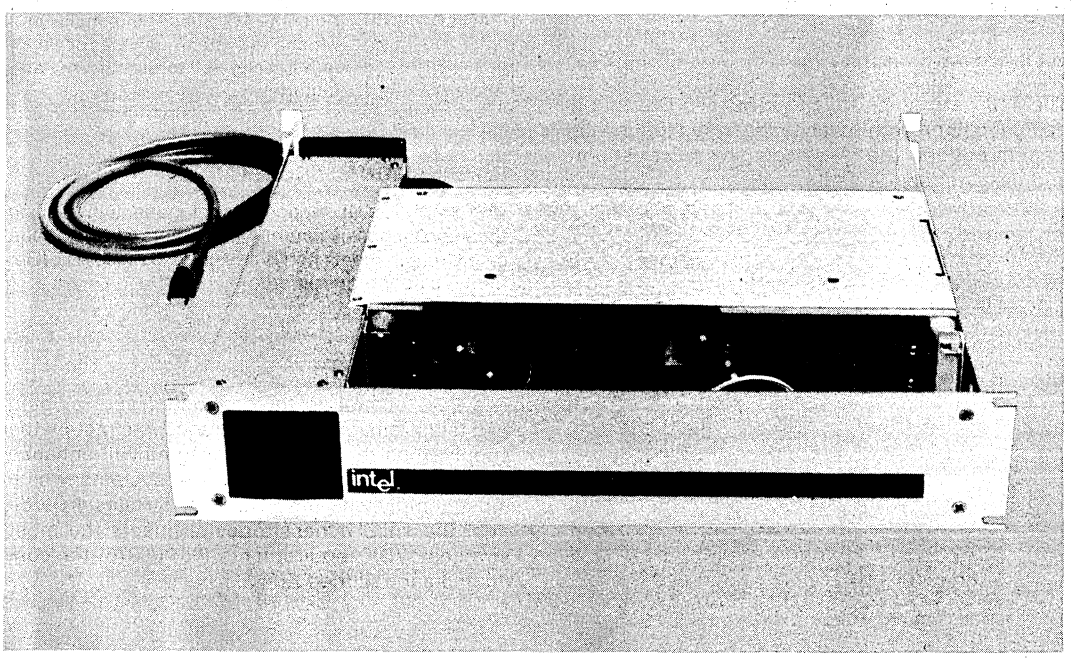
ORDERING INFORMATION

Part Number	Description
SBC 6611	Eight-slot MULTIBUS system chassis with parallel priority arbitration circuitry and 230 watt Linear Power Supply

iSBC® 665
SYSTEM CHASSIS

- **Intel MULTIBUS® system bus 4-slot packaging**
- **Complete package of rack-mounting, cooling, controls, and power**
- **Advanced 110 watt switching power supply generates ±5, ±12 VDC**
- **Meets U.S. and International EMI and safety requirements**
- **Wide AC voltage margins keep systems running during "brownouts"**
- **Front panel switches, indicators, and adjustments for operational and service convenience**
- **Power sense circuitry interrupts system 6 msec prior to power failure**

The Intel iSBC® 665 Chassis system provides the MULTIBUS system bus user with a compact set of products offering new standards in 4-slot rack-mount packaging. A high-efficiency switching power supply allows use of 115/230 VAC (+15%, −20%), with large surge and noise components, to deliver smooth, stable DC power to the OEM boad load. Advanced power-fail sense and restart logic gives the user sufficient time to bring the system to an orderly shutdown in the event of AC mains power failure. Mechanical design features include EMI suppression and a retainer/cover for system boards and I/O edge connectors.



April 1981
143581

FUNCTIONAL DESCRIPTION

iSBC® 665 System Chassis

The iSBC 665 Chassis is a complete microcomputer package providing four board slots in a 3.5″ vertical space.

RACK MOUNT PACKAGE

The iSBC 665 Chassis mounts in a 19″ EIA standard rack, using its front panel and separate support brackets at the rear of the chassis to secure it to both sets of rails in the cabinet. If slide mounting is preferred, a tray with slides should be used as a platform for the chassis. The physical integrity of the system is enhanced by addition of a connector retainer at the (rear-facing) opening of the cardcage.

INTEGRAL COOLING

The fan on the power supply is utilized to draw ambient air across the boards prior to its being used to cool the supply.

FRONT PANEL

The front panel of the iSBC 665 Chassis forms a complete control center for the system installed in the chassis (see Figure 1).

Power Supply (Fig. 1, 2)

The 110-watt supply of the iSBC 665 Chassis is designed to provide advanced features to the Intel system builder who faces complex power supply and chassis requirements.

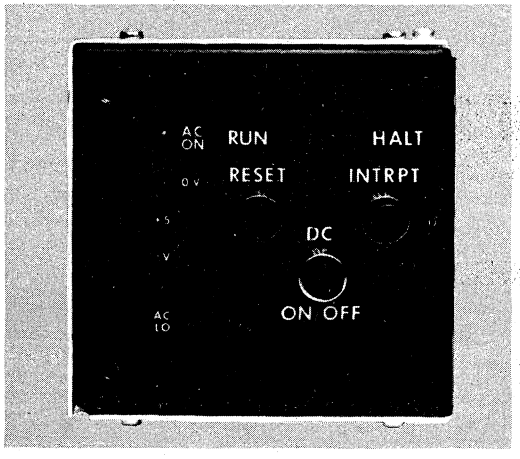


Figure 1. iSBC ® 665 Chassis Front Panel Controls

Table 1. iSBC® 665 Chassis Control Panel Functions (Fig. 1)

Label	Function
Controls	
DC ON/OFF	Controls all DC power to the chassis.
RESET	Generates RESET/ signal to pin 14 of P1 (MULTIBUS system bus) backplane.
INTRPT	Generates INT/ signal to pin 42 of P1.
Indicators	
HALT, RUN	Indicate status of system CPU board.
AC ON	Indicates AC power present in supply (AC power switch is located at the rear of the supply).
OV	Indicates power supply shut-down due to an overvoltage condition on +5 or ± 12 VDC outputs.
AC LO	Indicates that AC voltage is below the operating range and the supply has shut down.
Adjustments	
+5	Adjusts +5 VDC output voltage.
−V	Adjusts negative adjustable voltage; set to −5 VDC at the factory.
AC LO	Adjusts AC sense threshold at which the system generates power-fail signals; set to 88/176 VAC at factory.

INTERNATIONAL ACCEPTANCE

The supply is a UL-recognized component. In addition, the supply/chassis combination meets CSA (Canada) safety requirements and is designed to meet VDE (Germany) safety requirements.

EMI STANDARDS

The FCC standards for conducted and radiated EMI (electromagnetic interference) are met by the supply, thus the chassis packaging will enhance the OEM's efforts to assemble systems which must comply with the FCC Part 15 Rules. In addition, the supply/chassis design meets the most stringent VDE requirements (0871/0875) for conducted and radiated EMI.[1]

BROWNOUT PROTECTION

The wide AC voltage input range allows microcomputer systems packaged in the chassis to function normally at extremely low AC voltage supply levels.

POWER-FAIL WARNING AND RECOVERY

In the event of a complete power failure, an interrupt is generated 6 ms prior to the supply's issuing a subsequent memory protect signal, giving sufficient time for execution of a user program to bring the entire system to an orderly shut-down.

POWER TRANSIENT TOLERANCE

The supply provides immunity for the system from the high-voltage transient surges and spikes seen in AC power systems. The supply itself provides this isolation with a metal-oxide varistor (MOV) and line filter in the input circuitry.

POWER LINE CLOCK

A clock signal is developed from the AC line at twice the line frequency; this gives the system user an extremely accurate time base.

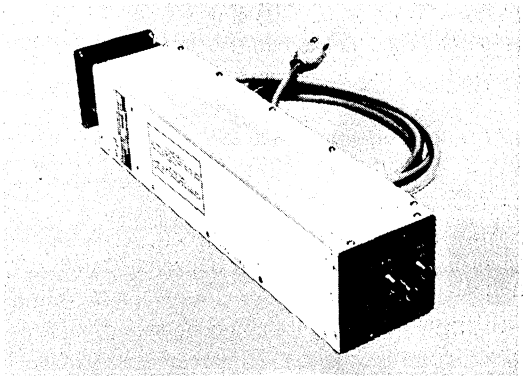


Figure 2. Power supply used on the iSBC® 665 Chassis

SPECIFICATIONS

Electrical Characteristics

INPUT POWER

Frequency: 47-66 Hz
Voltage: 115/230 VAC Single Phase
Range: 90 to 126 VAC/180 to 252 VAC
Consumption (Max.): 230 watts

OUTPUT POWER

Nominal Voltage	Current [2] (Max. Amps)	Current Limit Point (Amps)	Overvoltage Protection[3]
+5	15	18.75	5.25 to 6.25
+12	3	3.75	12.6 to 15.0
−12	1	1.25	−12.6 to −15.0
−Adjustable[4]	1	1.25	N/A

NOTES:
2. Total output power is 110 watts; a maximum of 128 watts is available, but proper operation of the power-fail circuitry is not guaranteed above 110 watts.
3. A minimum load is required on the +5 VDC output; this load must be at least ½ the sum of the loads (in watts) of the remaining three outputs.
4. −2.5 to −12 VDC; factory set to −5 VDC.

OUTPUT REGULATION (COMBINED LINE AND LOAD) — ±1% under any conditions of AC mains voltage variation (within operational range) and output load change.

PERIODIC AND RANDOM DEVIATION (PARD) — 50 millivolts peak-to-peak, all outputs.

LINE TRANSIENT TOLERANCE — A signal of up to 1000 VDC, with a pulse width of up to 50 microseconds, will have no affect on operation.

POWER FAIL INDICATION — An AC low condition generates ACLO and PFIN/ after AC voltage drops below the allowed voltage range. These signals are available on the P2 connector to generate interrupts. The DC voltages will remain within specifications for 6 milliseconds (worst case) following these interrupts, after which Memory Protect (MPRO/) will go true.

OUTPUT VOLTAGE TEMPERATURE COEFFICIENT — 0.03% per °C over the operating range.

SYSTEM CLOCK — 2× line frequency clock signal available on P2 connector.

BUS ARBITRATION — Serial; up to 3 CPU masters

Physical Characteristics (See Figure 3)

WIDTH — 19.0 in. (48.3 cm)
LENGTH — 16.25 in. (41.3 cm)
HEIGHT — 3.5 in. (8.9 cm)
WEIGHT — 12.0 lb (5.4 kg)

CARD SLOT SPACING — 0.6 in.

Environmental Characteristics

AMBIENT (INLET) AIR TEMPERATURE — Chassis: 0°C to 55°C; Power Supply: 0°C to 65°C (Full Rated Output)

HUMIDITY — Up to 95% non-condensing.

Equipment Supplied

iSBC 665 SYSTEM CHASSIS — Includes iSBC 645 Power Supply, iSBC 604 Modular Cardcage/ Backplane, connector retainer, schematics for cardcage/backplane, chassis, and power supply.

Reference Manual (Order Separately)

142836 — iSBC 665 System Chassis Hardware Reference Manual

Manual may be ordered from any Intel sales representative, distribution office, or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

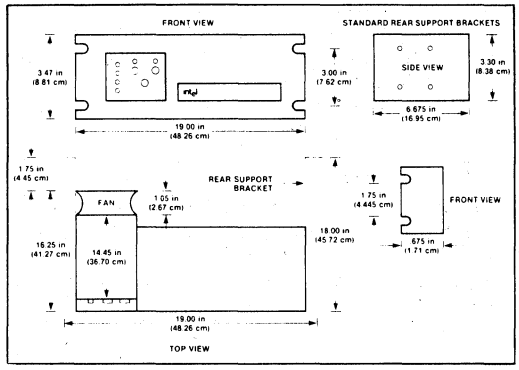


Figure 3. iSBC® 665 System Chassis Physical Dimensions

ORDERING INFORMATION

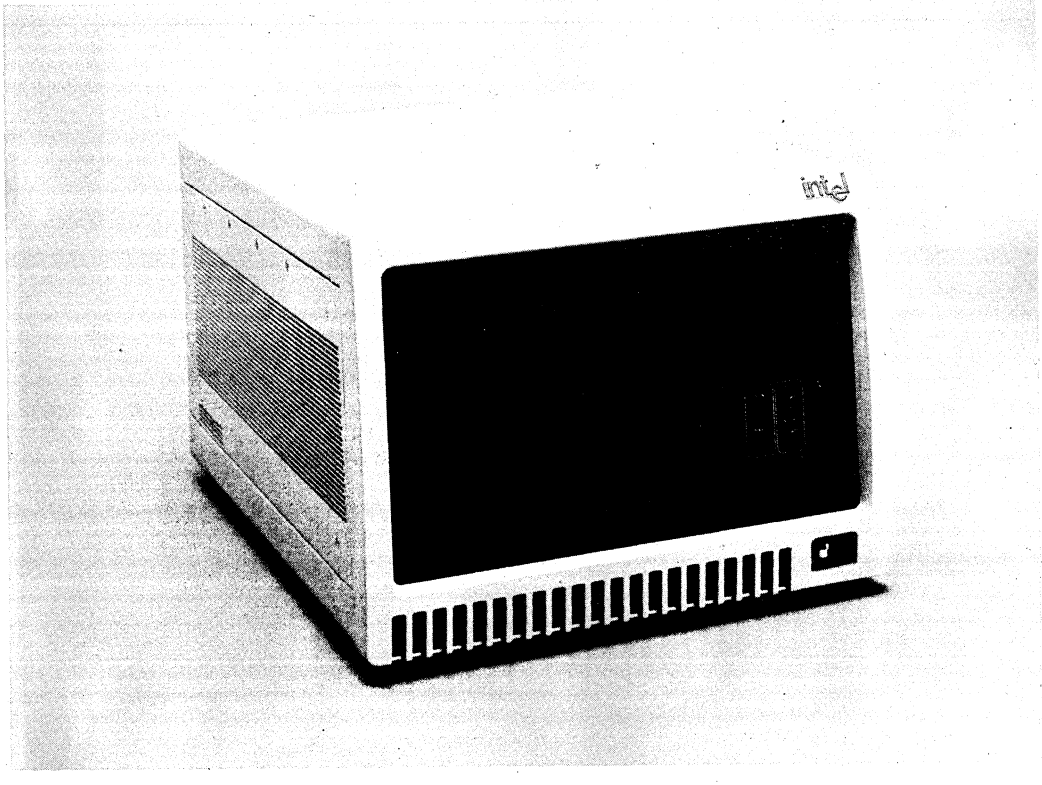
Part Number	Description
SBC 665	System Chassis

iSYP 384
SYSTEM CHASSIS

- **Fourteen-slot MULTIBUS® chassis with parallel priority circuitry and low noise/high speed backplane**
- **500 watt power supply provides 70 amps output at 5 volts**
- **Operates on 92-126 VAC, 47-63 Hz lines**
- **UL, CSA, FCC approved for data processing equipment — meets IEC-435**
- **Extra-wide cardcage slot spacing for iSBC® and iSBX™ MULTIMODULES™**
- **Fourteen connector ports for I/O cabling**

The iSYP 384 chassis is designed for the user who needs a reliable, easily configured chassis for high board count microcomputer applications. The iSYP 384 features a 14 slot cardcage, a 500 watt power supply, power cabling, system status controls and indicators, all fully integrated together in an attractively styled chassis.

The iSYP 384 chassis has many unique features which make it the best large system chassis on the market. First, the cardcage uses a parallel priority, low noise, high speed backplane. This allows the designer to install up to 7 CPU boards in the system and operate them at speeds up to 10 MB/sec. with full data integrity. Also for maximum reliability in the toughest environments, the chassis is cooled by three fans and powered by a conservatively rated, high efficiency switching power supply. Finally, the user can easily access the cardcage, power supply, I/O connectors and all chassis wiring by removing the pop-off top cover. This greatly simplifies initial system configuration, future upgrades, and field service.



JUNE 1984
ORDER NUMBER 280022-001

FUNCTIONAL DESCRIPTION

Mechanical Features

CHASSIS

The welded aluminum chassis is lightweight, strong, and rigid. To aid servicing and board/connector installation, the front, rear, top, bottom, and side covers are all removable. The chassis may be mounted on a bench top or installed in a rack, using an optionally available rack mount kit. The rack mount kit includes a wider front trim bezel to cover the mounting rails.

CARDCAGE

The 14 slot cardcage has 11 slots at 0.8″ spacing, which will accommodate MULTIBUS® boards with iSBC® MULTIMODULES™; and 3 slots at 1.2″ spacing, which will accommodate either the higher profile MULTIBUS boards with iSBX MULTIMODULES or prototype (wire wrapped) boards. All boards are held firmly in place with board retainers. The assembly uses an extruded, aluminum channel construction for strength, rigidity, and to ensure a solid electrical contact between the backplane and the MULTIBUS boards. Access to the cardcage is through the top cover of the chassis (see Figure 2).

COOLING

The chassis is cooled using three 100 CFM fans. A fully loaded chassis can be operated over a temperature range of 0° to −50°C (ambient).

I/O CONNECTORS

The removable back panel has 14 connector cutouts: 3 cutouts for 50 pin "D-ribbon" style connectors; 1 cutout for a 36 pin printer connector; and 10 cutouts for 25 pin, RS-232 type connectors. Each cutout has its own cover plate. Additional space is provided on the panel for the user to punch cutouts for any other size connector (see Figure 3).

Electrical Features

CHASSIS

The chassis is designed for domestic and international applications and will operate over a wide range of input voltages. The unit has been certified to meet

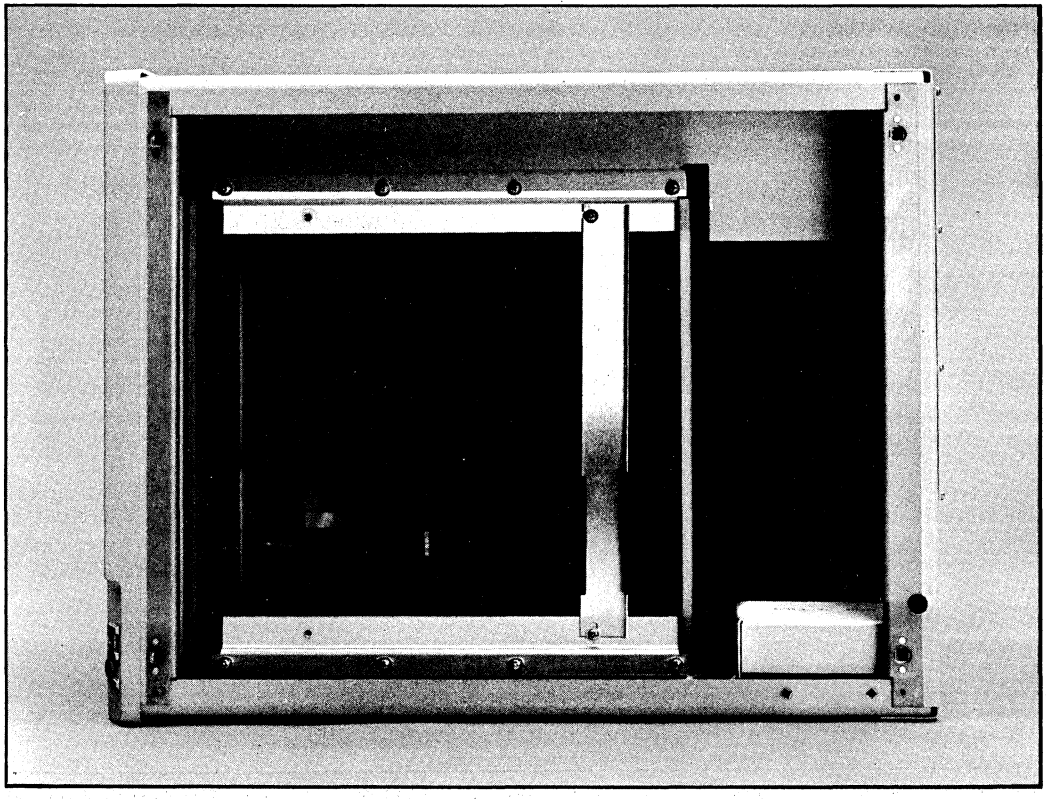


Figure 2. Cardcage in the iSYP 384 Chassis

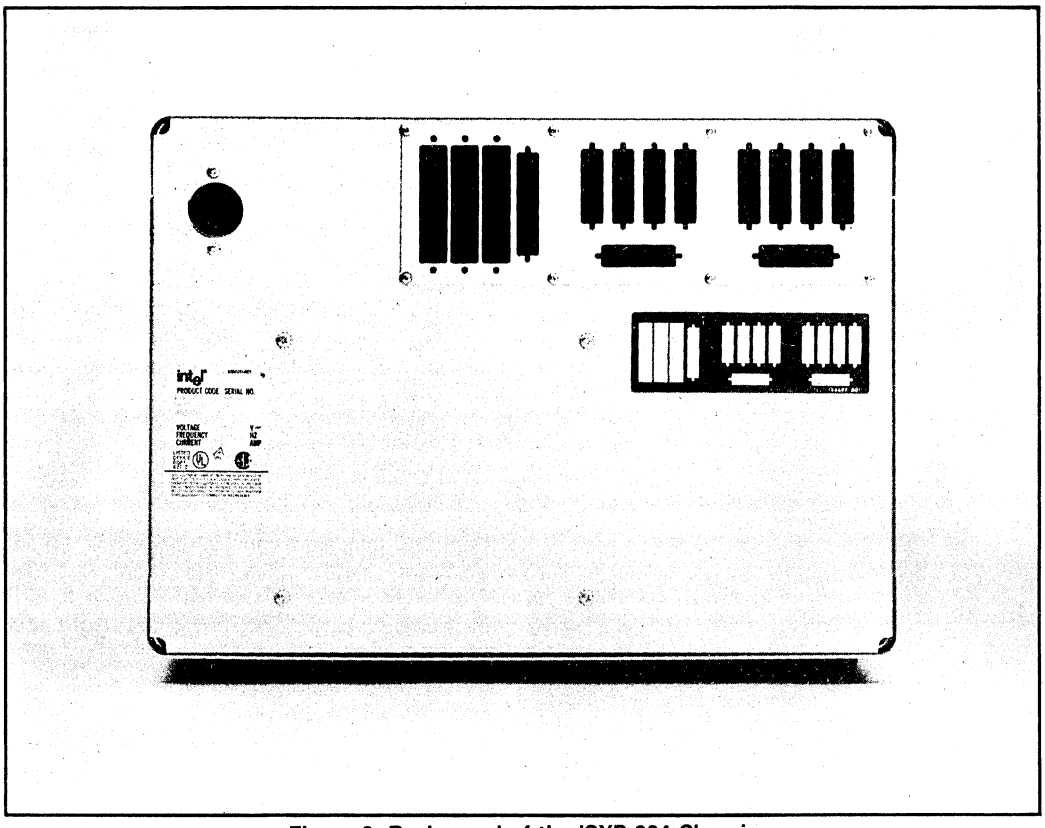


Figure 3. Back panel of the iSYP 384 Chassis

UL, CSA, and FCC requirements for data processing equipment, and has been designed to meet IEC-435 requirements. Three RFI filters, including one built into the power supply, keep conducted and radiated RFI/EMI emissions well below accepted limits. Furthermore, the filtering enables the chassis to operate reliably on the "noisy" power lines common in industrial environments.

POWER SUPPLY

The power supply is rated for 500 watts of power output, offering high current output over four DC voltages. All outputs are current limited and over-voltage protected.

A power fail circuit, incorporated in the power supply, measures the AC input voltage and generates an AC LOW signal during deep "brownout" or power fail situations. This signal, which is generated at least 10 msec before the DC voltages go out of regulation, can be used by the system software for memory protection and to execute an orderly shutdown.

CARDCAGE

The 14 slot cardcage/backplane is designed to ensure quiet operation of the bus. This is done by interleaving ground traces between all signal traces on the backplane and by providing a ground plane on the connector side of the board. The backplane uses a parallel bus arbitration scheme which allows up to 7 CPU boards to be installed in the system for multiprocessing applications. The priority of the board locations is set by jumpers on the backplane.

FRONT PANEL

On the front panel of the chassis are three LED indicators and two push button switches. The "POWER" LED monitors the +5V output and will remain on as long as both the AC input and the +5V output from the power supply are within limits. The RESET and INTERRUPT push button switches generate RESET/ and INT1/ signals on the MULTIBUS when depressed. The "RUN" LED (green) is on when the CPU is executing an instruction, and the "HALT" LED (red) is on whenever the processor executes a HALT instruction.

SPECIFICATIONS

AC Input Requirements

Input Voltage	Frequency	Max Current
90-132 VAC	47-63 Hz	10 A

Maximum power consumption is 1250W.

Output Power

Nominal Voltage	Max Output Current	Current Limits	Over-Voltage Protection
+5V	70.0A	143 to 168A	+10.5 to +11.5V
−5V	3.0A	6.2 to 7.2A	−10.5 to −11.5V
+12V	6.0A	12.3 to 14.4A	+25.2 to +27.6V
−12	5.0A	10.3 to 12.0A	−25.2 to −27.6V

The maximum power available from the supply, from all outputs, is 500 watts.

Product Safety Standards

The system is designed to meet UL standard 114 Safety of Electronics Data Processing Units and Systems; the Canadian Standards Association standard C22.2 154-1975 Safety of Data Processing Equipment; the applicable RFI/EMI requirements of VDE 0871/6.78, VDE 0875/6.77; and FCC rule 47 CFR part 15 subpart J Emission Limits for Computing Devices.

Environmental Requirements

OPERATING

Temperature: 0°C to 50° C

Relative Humidity: 10% to 85% non-condensing over the operating temperature range. The environmental combination of humdity and temperature cannot exceed 26°C wet bulb.

NON-OPERATING

Temperature: −40°C to 70°C

Relative Humidity: 20% to 80% non-condensing

Vibration: .020 inches, peak-to-peak, 5-25 Hz; .010 inches, peak-to-peak, 25-65 Hz; 2.0g, 0-to-peak, 65-300 Hz

Physical Characteristics

Width: 16.8 in. (42.6 cm)

Height: 12.2 in. (31.1 cm)

Depth: 21.0 in. (53.3 cm)

Weight: 55 lb. (25 kg)

REFERENCE MANUALS (Not included: order separately)

System 86/380 Hardware Reference Manual
Order Number: 172761-001

Reference manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department.

ORDERING INFORMATION

ORDER CODE

iSYP 384-7

DESCRIPTION

Fourteen-slot MULTIBUS system chassis with parallel priority bus arbitration circuitry and 500 watt switching power supply, 90-132 VAC, 47-63 Hz

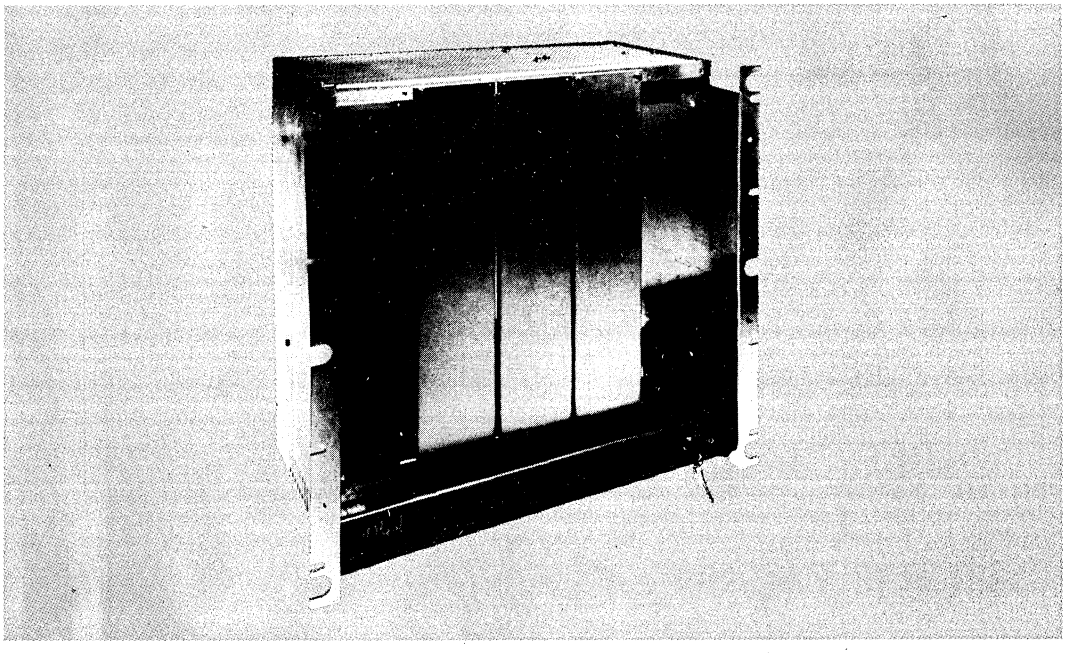
ICS Industrial Control Series and Analog I/O Expansion

13

iCS™80 INDUSTRIAL CHASSIS KIT 635, KIT 640

- **Available with iSBC® 635 or iSBC® 640 power supply**
- **Accommodates from 1 to 3 iSBC® 604/614 cardcage assemblies for 4-12 MULTIBUS® board capacity**
- **Vertical board orientation and four fans for high efficiency cooling**
- **Front access to iSBC® boards, power supply, and signal conditioning panels**
- **19-inch wide RETMA rack mounting or NEMA type backwall mounting brackets**
- **UL and CSA approved**
- **Multi-voltage operation**
- **Lockable service panel**
- **Recessed mounting space for signal conditioning/wire termination panels**

The iCS 80 Industrial Chassis provides industrially oriented mounting space for Intel single board computer (iSBC) products, associated iSBC power supplies, and related iCS 9XX analog and digital conditioning/termination panels. The base unit provides a 4-slot MULTIBUS backplane (iSBC 604) with expansion space and cabling to expand to 12 MULTIBUS backplane slots by adding additional 4-slot iSBC 614s as needed (up to two). Full MULTIBUS compatibility in the iCS 80 chassis allows configuration of multiple single board computers to share system tasks through communication over the bus (through multimaster bus arbitration built on the multiple iSBC processors).



FUNCTIONAL DESCRIPTION

iCS™ 80 Kit 635

Small, self-contained industrial controllers can be configured with the 4-slot cardcage and iSBC 635 power supply. As shown in Figure 2, this chassis can also accommodate the iCS 9XX series signal conditioning termination panels.

iCS™ 80 Kit 640

This chassis uses the higher power iSBC 640 power supply, and is designed to power higher board count systems. By installing one or two additional iSBC 614 cardcages, this chassis will accommodate up to 8 or 12 MULTIBUS boards.

Engineered for Industrial Applications

The MULTIBUS slots are mounted vertically to improve convection cooling and the top, bottom and sides are engineered to allow maximum air flow over the boards. Four fans are provided to increase air flow, allowing users to eliminate or minimize the need for supplementary fans or air conditioning.

Power Supply Flexibility

The power supplies are mounted on slide in/out mounting rails, and quick disconnect cabling and connectors are provided for rapid service replacement. An AC wiring barrier strip allows simple wiring connections for integration into larger systems (see Figure 4).

Industrial Rack Mounting

The chassis mounts directly into 19-inch standard width RETMA (Radio-Electronics-Television Manufacturers Association) customer provided rack. Alternately, mounting brackets and power cabling access are provided for mounting directly on a backwall, such as the backwall panel of a NEMA-type (National Electrical Manufacturers Association), front-access-only cabinet.

Front Access Serviceability

To simplify serviceability, front access is provided for all iSBC boards, the power supply, operation indicator lights, interrupt and reset buttons, and the AC power fuse.

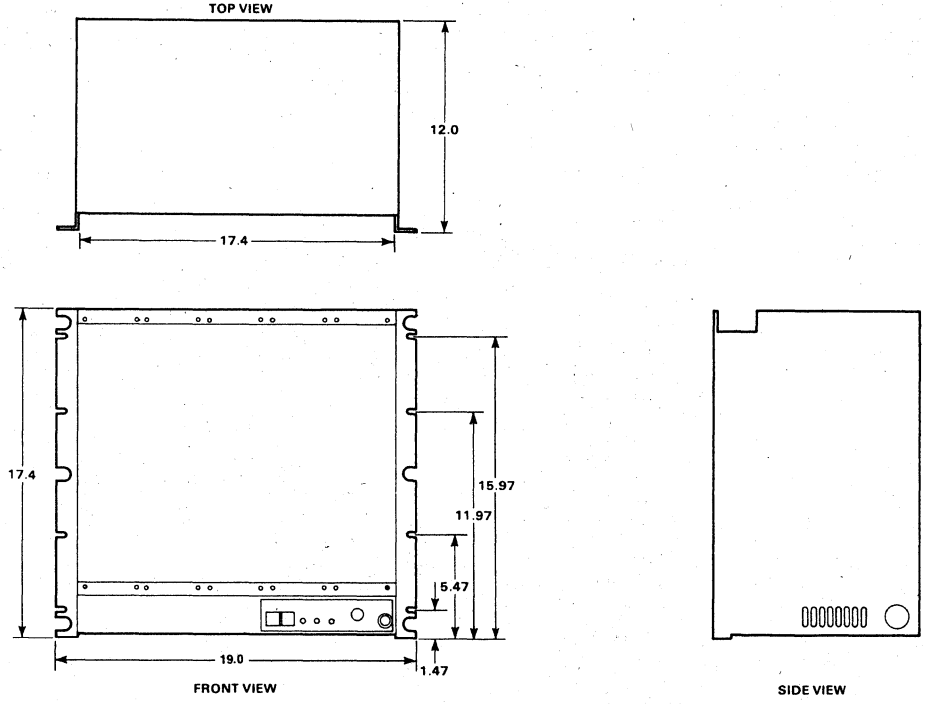
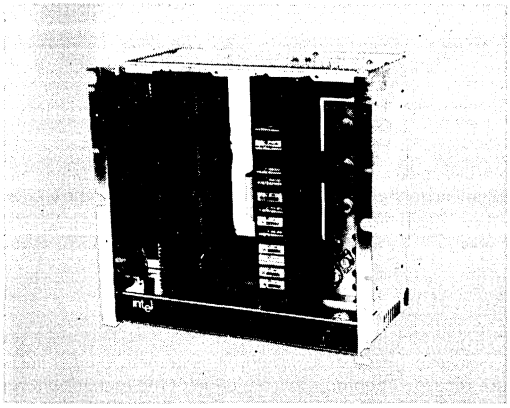


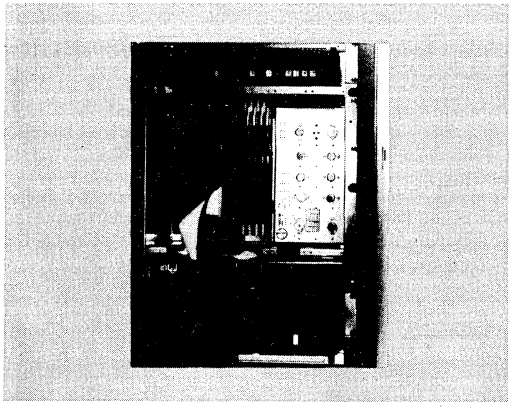
Figure 1. iCS™ 80 Chassis Dimensions



Typical Small Configuration

- iSBC 88/40A Test and Measurement Computer
- iCS 910 Analog I/O Signal Conditioning Panel
- iCS 930 AC/DC Control Interface Panel

Figure 2. Small Configuration iCS™ 80 Kit.



Typical Maximum Configuration

- 16-bit 8086 processor (iSBC 86/30 w/RAM MULTIMODULE)
- 768K bytes RAM (2 - iSBC 056A)
- 128K bytes EPROM (or 16K E^2PROM)
- 240 analog inputs (3 - iSBC 88/40A w/2 ea. iSBX 311)
- 24 analog voltage outputs

OR

- 24 analog current outputs (4-20 mA)
- 72 isolated digital inputs/outputs
- 144 TTL digital inputs/outputs (2 - iSBC 519s)

(All iCS 9XX Signal Conditioning/Termination Panels shown mounted to cabinet)

Figure 3. iCS™ 80 Kit 640 with 12 MULTIBUS® Card Slots Mounted in NEMA Cabinet

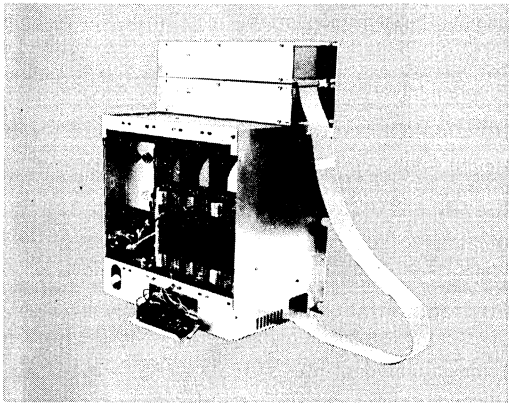


Figure 4. Rear View iCS™ 80 Chassis Showing Power Distribution Panel (detached to show terminal block), and Cabling from iCS 80 Chassis to iCS 9XX RETMA Mounted Signal Conditioning Panels (Top of iCS 80 Chassis)

Lockable Service Panel

To assist in development, checkout and service, two pushbuttons are provided. The RESET button pulls low the initialize line (INIT) on the MULTIBUS backplane. The INTERRUPT button pulls low one interrupt line on the MULTIBUS backplane (INT1). Logic within the iCS 80 ensures that these buttons function with all versions of Intel single board computers. From the front of the iCS 80 chassis, without a CRT or other panel, an operator or service person can reset or interrupt on-going iCS 80 system operations to get attention, signal an alarm, or start a self-test operation.

A front panel key provides three positions: OFF (AC power off and key removable), ON (AC power on, pushbuttons enabled, key unremovable), and LOCK (AC power on, pushbuttons disabled, key removable).

Three indicator light emitting diodes record basic chassis status. POWER ON (GREEN); RUN (GREEN); and HALT (RED); the RESET or INTERRUPT buttons will remove the HALT state.

U.L. Approved

The iCS 80 chassis has received full Underwriters Laboratory approval (F.6 #E70842) as a U.L. listed component under the Underwriters Laboratories Safety Standard for Process Control Equipment, UL1092. When installed as described in the iCS 80 Hardware Reference Manual, the iCS 80 chassis provides adequate protection against shock, fire and casualty hazards, and should comply with most local and regional requirements for installation in ordinary locations. In addition, the iCS 80 chassis was designed to comply with the UL requirements for Data Processing Equipment, UL478. The iCS 80 has also been approved by the Canadian Standards Association under CSA category C22.2 No. 142, the Canadian Standard for Safety for Process Control Equipment and C22.2 No. 154 for Data Processing Equipment.

Mounting Space for Signal Conditioning/Wire Terminations

The cardcages and power supplies in the iCS 80 chassis are recessed behind the front edge of the rack mounting ears to provide mounting space for the iCS 9XX series signal conditioning/termination panels and field wiring. For smaller systems with only one or two iSBC 604/614 cardcages (4 to 8 slots), up to two iCS 910, iCS 920, or iCS 930 signal conditioning/termination panels can be mounted vertically over the area where the second or third cardcage would mount (see Figure 2). The benefit of this design is a completely self-contained industrial chassis with iSBC cards, power supply, signal conditioning and field wiring terminations, all in one enclosure.

SPECIFICATIONS

Capacity

Four slots for MULTIBUS compatible single board computers, memory, I/O or other expansion boards
Expandable to 12 slots using two iSBC 614 cardcages (Order Separately

Front Panel Controls

Pushbuttons

RESET: Connected to Initialize/ on MULTIBUS backplane

INTERRUPT: Connected to Interrupt 1/ line on MULTIBUS backplane.

Panel Indicator Lights (LEDs)

POWER ON (green): +5V power exists on the MULTIBUS backplane

RUN (green): CPU is executing an instruction. Light goes out if CPU is in WAIT or HALT state

HALT (red): CPU has executed a HALT instruction

Keylock

OFF: AC power off, key removable

ON: AC power on, pushbuttons enabled, key unremovable

LOCK: AC power on, pushbuttons disabled, key removable

Fuse — AC power (6A)

Equipment Supplied

iCS 80 industrial chassis, three fans for cardcages, one fan for power supply, 4-slot cardcage with MULTIBUS backplane, control panel with switches, indicators, keylock, power distribution barrier strip, AC power fuse, line filter, 115V power cable, and logic for interrupt and reset buttons. An installation package is also provided, including a NEMA cabinet mounting kit, power supply extension cables, and RETMA cabinet mounting screws, 100/120/220/240 VAC operation.

Software

See the RMX/80 Real-time Multitasking Executive specifications for industrial related applications. In addition, system monitors for most of the Intel single board computers are available in the INSITE (Intel's Software Index and Technology Exchange) User's Program Library.

Physical Characteristics

Height — 39.3 cm (15.7 in.)
Width — 48.5 cm (19.0 in.) at front panel
43.5 cm (17.4 in.) behind front panel
Depth — 30.0 cm (12.0 in.) with all protrusions
Weight — 16.8 kg (37.0 lb) without power supplies

Environmental Characteristics

(Ambient at iCS-80 air intake, bottom of chassis)

Temperature (Ambient)
Operating: 0°C to 50°C (32°F to 122°F)
Non-operating: −40°C to +85°C

Humidity — Up to 90% relative, noncondensing at 40°C

Electrical Characteristics

The iCS 80 chassis provides mounting space for either the iSBC 635 or iSBC 640 power supply. Unless otherwise stated, electrical specifications apply to both power supplies when installed by user in iCS 80 chassis.

Input Power

Frequency: 47 to 63 Hz. Voltage (Nominal)
Voltage (Single Phase, Jumper Selectable)
iCS 80 Kit 635: 100, 115, 215, 230 VAC (±10%)
iCS 80 Kit 640: 100, 120, 220, 240 VAC (±10%)

	With ISBC 635	With ISBC 640	Input Voltage
Current: (Including fans)	3.0A max	5.6A max	103 VAC
	1.5A max	2.8A max	206 VAC
Power, max:	315 watts	580 watts	

Output Power

Voltage	Output Current (max)		Overvoltage Protection	
	iSBC 635	iSBC 640	iSBC 635	iSBC 640
+ 12V	2.0A	4.5A	+ 14V to + 16V	+ 14V to + 16V
+ 5V	14.0A	30.0A	+ 5.8V to + 6.6V	+ 5.8V to + 6.6V
− 5V	0.9A	1.75A	− 5.8V to − 6.6V	− 5.8V to − 6.6V
− 12V	0.8A	1.75A	− 14V to − 16V	− 14V to − 16V

Combined Line/Load Regulation — ±1% at ±10% static line change and ±50% static load change, measured at the output connector (±0.2% measured at the power supply under the same conditions).

Remote Sensing — Provided for +5 VDC output line regulation.

Output Ripple and Noise — 10 mV (iSBC 635 and iSBC 640 supply) peak-to-peak, max (DC to 500 kHz)

Output Transient Response — Less than 50 μsec for ±50% load change.

Maximum Watts Dissipation (load plus losses) — 500W (iSBC 640 supply), 250W (iSBC 635 supply)

Installation

Complete instructions for installation are contained in the iCS 80 Site Planning and Installation Guide, including RETMA and NEMA cabinet mounting, and field signal, ground wiring and cooling suggestions.

Warranty

The iCS 80 Industrial Chassis is warranted to be free from defects in materials and workmanship under normal use and service for a period of 90 days from date of shipment.

Reference Manuals

9800799A — iCS 80 Industrial Chassis Hardware Reference Manual (SUPPLIED)

9800708A — iSBC 604/614 Cardcage Hardware Reference Manual (SUPPLIED)

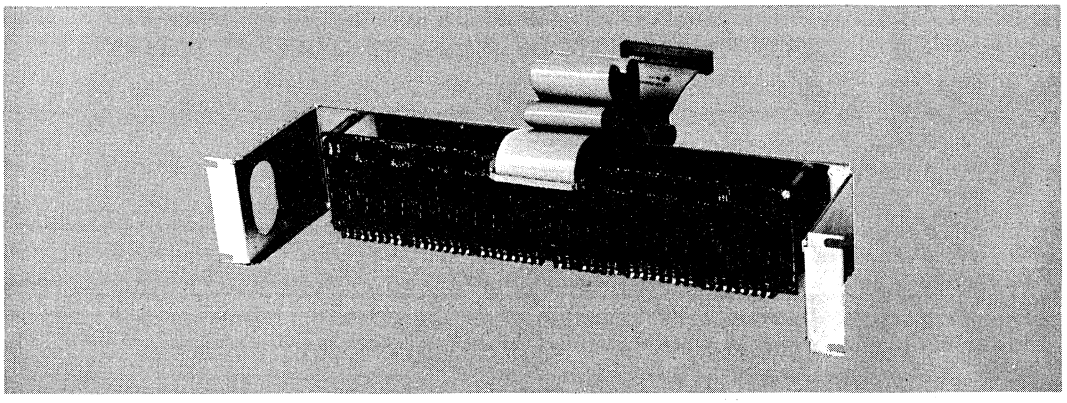
ORDERING INFORMATION

Part Number	Description
ICS 80 Kit 635	iCS 80 system consisting of: iCS 80 Industrial Chassis iSBC 635 Power Supply
iCS 80 Kit 640	iCS 80 system consisting of: iCS 80 Industrial Chassis iSBC 640 Power Supply

iCS™ 910/920/930
SIGNAL CONDITIONING/TERMINATION PANELS

- Interconnects iSBC ® and digital I/O ports to field signal/control wiring
- Ribbon cable connection from panel is pin compatible with iSBC ® analog, CPU, and digital board I/O ports
- Barrier strip screw terminals for
  - iCS 910: 32 single-ended analog inputs (or 16 differential signal plus shield) plus four analog voltage outputs or two analog 4 to 20 mA current outputs
  - iCS 920: 24 medium power digital inputs and/or outputs (55V, 300 mA max)
  - iCS 930: 16 high power AC or DC digital inputs or outputs (280 VAC, 3A max)
- Flexible mounting kits for
  - 19″ width RETMA rack
  - NEMA type backwall
  - iCS 80 Industrial Chassis
- Digital signal conditioning (iCS 920/930)
  - Sockets for optically isolated input filters and solid state output switches
  - Pad space for transient suppressors, current limiting resistors, and voltage dividers
  - Socketed fuse for overload protection (iCS 930)
  - LED/channel status indicators
- Engineering printed circuit mounting space for customer analog input components (iCS 910)
  - Noise fiters
  - Current loop resistors
  - Open circuit detection resistors
  - Voltage divider resistors
  - Thermistor bias current
- UL listed component

The iCS 910/920/930 Signal Conditioning/Termination Panels are heavy duty printed circuit boards with screw terminations which allow industrial customers to easily connect their heavier gauge field signal wiring to Intel's line of 8- and 16-bit single board computers, and iSBC analog and digital I/O boards. Flat, 50 conductor ribbon cables connect the iCS 910/920/930 panels to the 50 pin analog/digital I/O ports of Intel's MULTIBUS® boards and MULTIMODULES™. Power for opto-isolators or line drivers (+5 VDC) can be supplied via this cable from the iSBC boards. Jumpers and a screw terminal block are provided on the iCS 920/930 panels to allow an external supply of +5V power. A similar jumper/terminal block is provided on the iCS 910 panel to allow users to connect external +15V (or greater) compliance voltage for larger analog output loads.



FUNCTIONAL DESCRIPTION COMMON TO iCS™ 910/920/930

Large Wire or Spade Lug Connections

The barrier strip screw terminations on the iCS 910/920/930 panels provide familiar connection points for factory electricians to terminate the heavier gauge wiring often pulled through conduits from sensors or control elements. These screw terminals securely connect up to 14 AWG gauge wire size (16-gauge on iCS 910/920 panels). Alternately, spade lugs can be crimped on field wiring and inserted under the screw terminals.

Mounting Flexibility and Serviceability

The iCS 910/920/930 panels were designed to be physically separate from iSBC boards or the iCS 80 chassis to allow maximum mounting flexibility and ease of serviceability. The panels and field wiring can be mounted in one area of the cabinet where electricians have access. Flat ribbon cable can then be run to the area where control electronics technicians have access.

The iCS 910/920/930 panels may be mounted horizontally in a 19" standard width (RETMA) rack using a recessed mounting panel (see Figure 1). Alternately, the panels can be mounted on a cabinet wall (e.g., NEMA cabinet backwall) using standoffs provided (see Figure 2). Or, for the most compact packaging, users can mount up to two iCS 910/920/930 panels vertically, directly on the front of the iCS 80 chassis using standoffs and holes provided (see Figure 3).

A black metal labelling strip is provided with each iCS 910/920/930 panel. White, blank gummed labels are included so that users can custom identify each input or output channel. A clear plastic cover is provided to protect against inadvertent touching or damage to the screw terminals or customer mounted components.

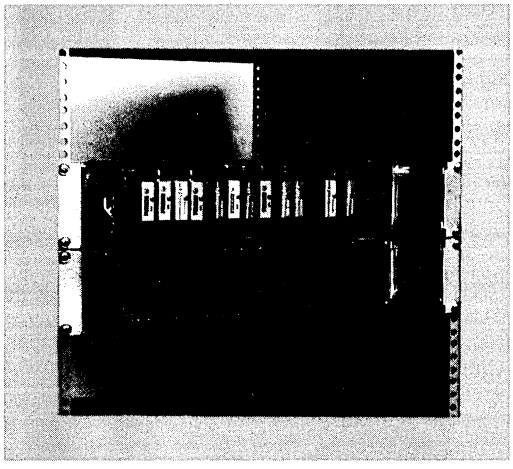


Figure 1. iCS™ 930 AC and iCS™ 920 Digital Signal Conditioning/Termination Panel Mounted on a 19" Width RETMA Rack

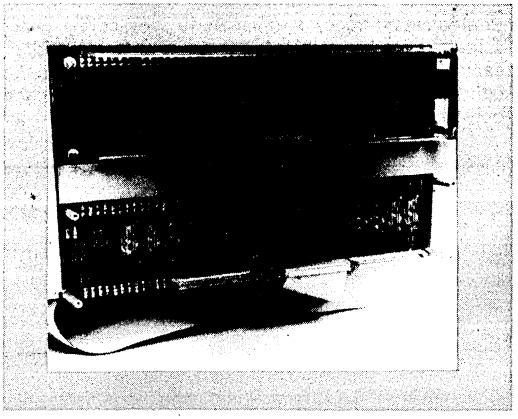


Figure 2. iCS™ 910/920 Signal Conditioning/Termination Panel Mounted on a NEMA Cabinet Backwall

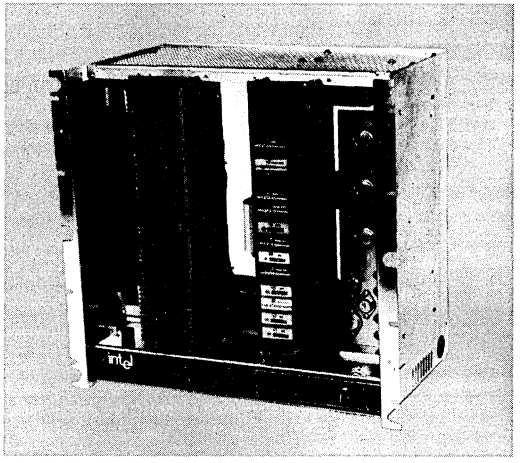


Figure 3. iCS™ 910/930 Signal Conditioning/Termination Panels Mounted on iCS™ 80 Industrial Chassis

iCS 910 ANALOG SIGNAL CONDITIONING/TERMINATION PANEL

Mixed Analog Input and Output Signals

A single iCS 910 panel connects up to 32 single ended analog inputs (or 16 differential analog inputs plus shield) to the iSBC 88/40A measurement computer or iSBX 328 A/O MULTIMODULE. In addition the same iCS 910 panels can connect up to four analog output voltages from the iSBX 311 analog output board. Three flat ribbon cables are included in the iCS 910 installation kit (two analog inputs, one analog output).

Engineered Signal Conditioning Mounting Space

Printed circuit traces on the iCS 910 panel connect each screw terminal analog input channel to the flat ribbon cable connector. Users can jump straight through signal connections if they desire. Each input channel trace, however, passes through a custom engineered printed circuit area onto which users may mount components to signal condition analog input signals. Pad traces and holes are designed to allow easy mounting of R-C noise filters, input voltage resistor/divider networks, current loop input resistors, open circuit detection resistors, or to supply thermistor bias current (see Figure 4 for schematic of a typical analog input channel).

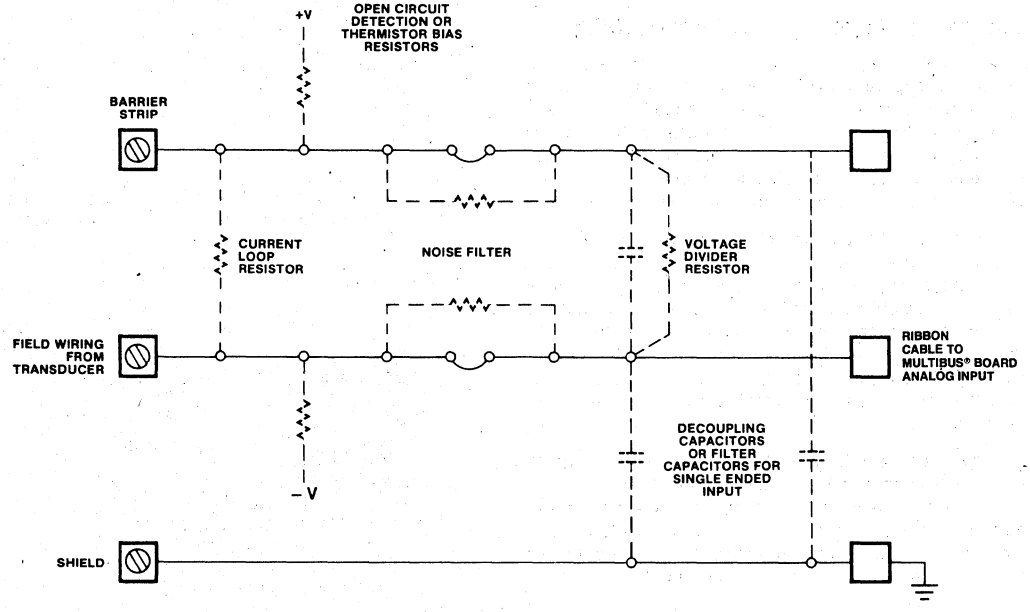


Figure 4. iCS™ 910 Analog Input Signal Conditioning Examples

iCS™ 920 DIGITAL SIGNAL CONDITIONING/TERMINATION PANEL

The iCS 920 panel interconnects up to 24, 2-wire digital input or output channels from barrier strip screw terminals to the 16- or 24-bit digital I/O ports, standard on many Intel single board computers and digital I/O expansion boards. Screw terminals allow for one each 16 AWG size wire for differential (2-wire) connections or two each AWG 18-gauge wire for daisy chaining grounds or power for external contact sensing.

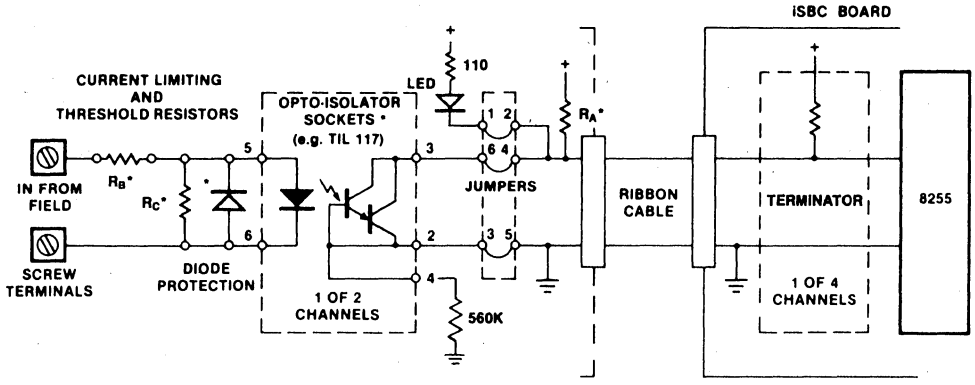
Flexibility in Isolation and Serviceability

Dual-in-line sockets are in series with each channel (see Figure 5) to allow customer jumpering for straight through connections (TTL I/O), or for insertion of popular DIP packaged opto-isolators or digital output high current driver transistors. Circuit pads are available for mounting voltage divider/threshold resistors and protection diodes.

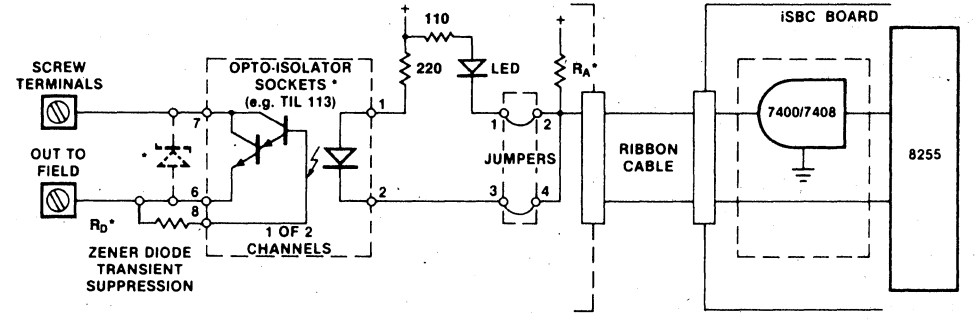
Groups of four inputs can have mixed voltage levels, opto-isolation, or straight through connections in groups of two. Output groups of four can be mixed opto-isolated or high current drive in groups of two. DIP components from a wide variety of vendors are selected and inserted by users based on their application. The iCS 920 manual recommends several alternative components and offers design assistance for your I/O configuration. Digital signal conditioning examples for several common industrial voltages are shown in Table 1 and in the diagrams below (see Figure 5).

Active Channel Indicators

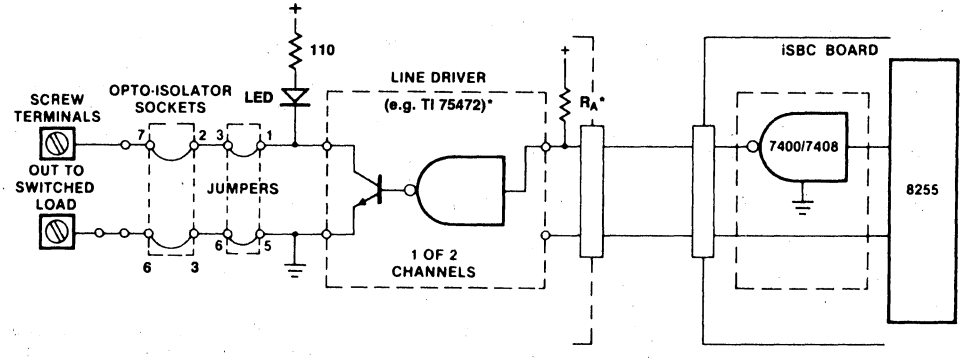
Light emitting diodes (LEDs) are mounted adjacent to each channel's screw terminals and may be jumpered in to indicate the Hi-Lo status of each of the 24 input or output channels.



OPTICALLY ISOLATED DC INPUT EXAMPLE (iCS-920 panel)



OPTICALLY ISOLATED DC OUTPUT EXAMPLE (iCS-920 panel)



CURRENT DRIVER OUTPUT (55V, 300 mA) EXAMPLE (iCS-920 panel)

*USER SUPPLIED COMPONENT

Figure 5. Digital Signal Conditioning Examples

AFN-01277A

Table 1. iCS™ 920 Digital I/O Signal Conditioning Plug-In Component Examples

Digital Voltage Input or Output Load Voltage	Maximum Input Current (mA)	Threshold Voltage (V)	Opto-Isolators*	Diode Protection*
Opto-Isolated Input				
5 VDC	50	3	TIL117	1N4002
12 VDC	50	6	TIL117	1N4002
24 to 26 VDC	40	6	TIL117	1N4002
48 VDC	20	12	4N36	1N4002

	Maximum Output Current (mA)	Line Driver*	Opto-Isolators*
Opto-Isolated Output			
12 VDC	100	—	TIL113
24 VDC	100	—	TIL119
48 VDC	100	—	MCS 2
Current Drivers			
55 VDC	300	TI75472	—
Half Wave Rectifier Outputs			
24 VAC SCR	300	—	GE4N40
115 VAC SCR	150	—	MCS 2

*Example component — alternate source components are listed in the iCS 920 Hardware Reference Manual.

iCS™ 930 AC Signal Conditioning/Termination Panel

The iCS 930 panel interconnects 16 2-wire digital input or output channels from barrier strip screw terminals to 16 bits of the digital I/O ports available on many Intel single board computers and digital I/O expansion boards. The iCS 930 panel differs from the iCS 920 digital signal conditioning/termination panel in that the iCS 930 panel handles higher AC or DC voltages and currents (up to 280V, 3A), such as those found on many 115 VAC machines, motor starters, and industrial control panels. The iCS 930 panel is also recommended for optically isolated DC outputs greater than 100 mA.

The iCS 930 screw terminals accept up to 14 AWG size wire each for differential (2 wires per channel) connections, or two 14 AWG size wires for daisy chaining grounds or power from external sources.

Modular Isolation/Switching with Easy Serviceability

Each iCS 930 panel accepts up to 16 user supplied, optically isolated input modules or optically isolated solid state switches, for either AC or DC voltages (see Figure 6). Each module is screw mountable/replaceable and can be mixed for AC or DC input, or AC or DC output, in groups of four. Among groups of four inputs (or outputs) each channel can be individually mixed for AC or DC input (or AC or DC output). The user pays only for those channels implemented. User supplied compatible modules are shown in Table 2.

DC and AC input modules are current actuated and thus provide a 5-ms filter against spurious noise spikes or contact bounce. AC solid state output modules provide zero crossing turn on to minimize arcing.

Protection Circuitry

Each of the 16 channels contain a socketed fuse to protect against overload. In addition, mounting pads are available on each channel output for user supplied voltage transient RC "snubber" components or inductive pulse suppression, e.g., metallic-oxide-varistor (MOV) for large motor starting.

Active Channel Indicators

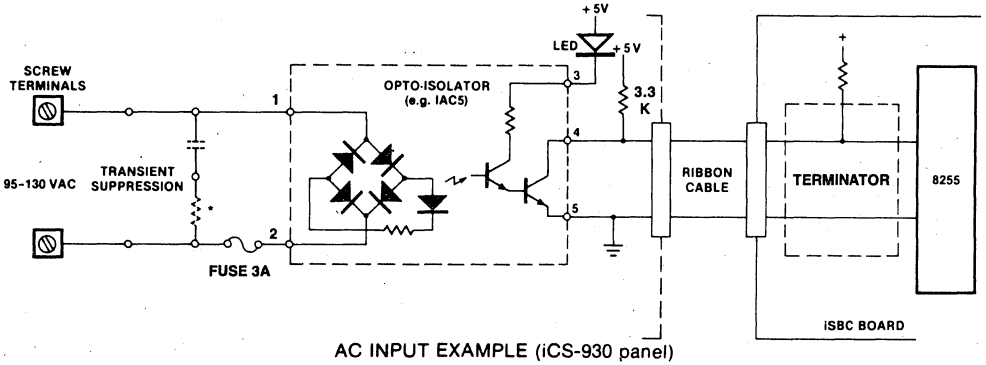
Light emitting diodes (LEDs) are mounted adjacent to each channel's screw terminals and opto-module to indicate Hi-Lo status of that channel and to assist in troubleshooting servicing.

Examples of iCS 930 input and output schematics are shown in Figure 6.

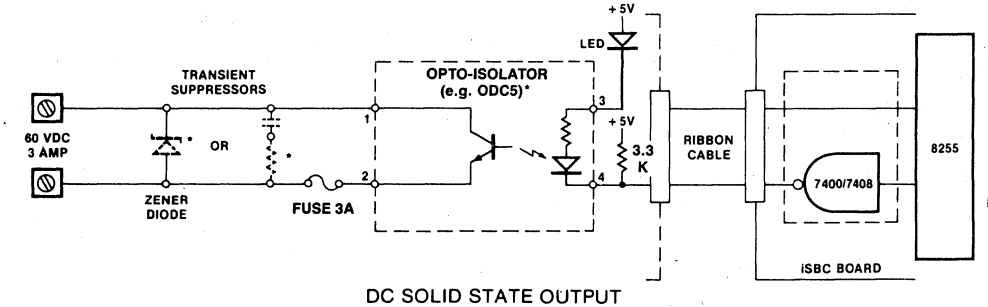
Table 2. Optical Isolated Modules Compatible with ICS™ 930 Signal Conditioning/Termination Panel

Signal Conditioning Desired	Voltage Rating	Maximum Input Current	Opto-22 Number*	Motorola Number*
AC Input — 115 VAC	95 to 130 VAC	10 mA	IAC5	IAC5
220 VAC	180 to 280 VAC	10 mA	IAC5A	
DC Input — 5 μsec Filter	10 to 32 VDC	32 mA	IDC5	IDC5
Fast, 50 μsec On	4 to 16 VDC	14 mA	IDC5B	
		Output Current Rating		
AC Output	12 to 140 VAC	3A	OAC5	OAC5
	24 to 280 VAC	3A	OAC5A	
DC Output	10 to 60 VDC	3A	ODC5	ODC5
	200 VDC	1A	ODC5A	

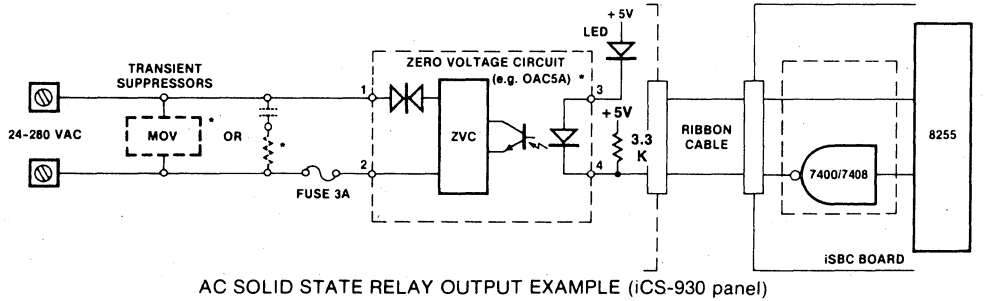
*Motorola and Opto-22 sales offices are located in North America, Europe, and Japan.



AC INPUT EXAMPLE (iCS-930 panel)



DC SOLID STATE OUTPUT



AC SOLID STATE RELAY OUTPUT EXAMPLE (iCS-930 panel)

*USER SUPPLIED COMPONENT

Figure 6. Typical ICS™ Signal Conditioning Examples

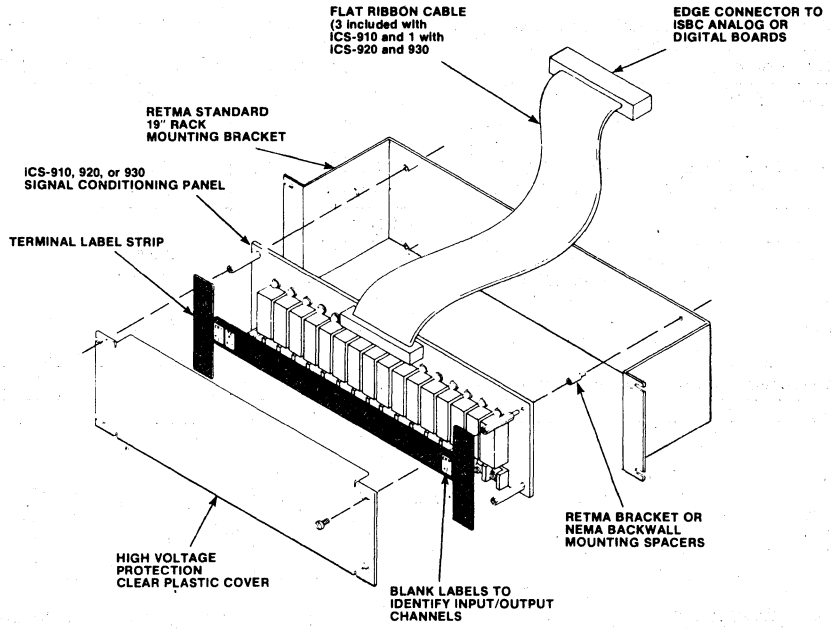


Figure 7. Mounting Arrangements for Signal Conditioning/Terminal Panels

SPECIFICATIONS

(For iCS 910/920/930 panels unless otherwise specified)

Number of Lines

iCS 910 Panel

Analog Inputs — Sixteen 3-wire (differential signal plus shield) or 32 single ended

Analog Outputs — Four 2-wire voltage output or two 2-wire current output

iCS 920 Panel — Zero to 24 digital inputs or outputs in groups of four

iCS 930 Panel — Zero to 16 digital inputs or outputs in groups of four

Isolation Characteristics

Line-to-Line Isolation — 250 VDC or RMS AC (iCS 910/920 panels), 500 VDC or RMS AC (iCS 930 panel)

Input/Output Isolation — 250 VDC or RMS AC (iCS 920 panel), 500 VDC or RMS AC (iCS 930 panel)

Physical Characteristics

Width: 36.63 cm (14.65 in.)
Height: 8.13 cm (3.25 in.)
Thickness: 0.24 cm (0.093 in.), iCS 910/920 panel
0.32 cm (0.125 in.), iCS 930 panel

	iCS 910	iCS 920	iCS 930
Weight:	(Minimum, PC panel only)		
	455 gm(16 oz)	455 gm (16 oz)	681 gm (24 oz)
	(Maximum with all components and mounting kit installed)		
	1.6 Kg (56 oz)	1.8 Kg (64 oz)	3.4 Kg (120 oz)
Depth:	(With components and clear plastic cover installed)		
	5.08 cm(2.0 in.)	5.08 cm (2.0 in.)	5.08 cm(2.0 in.)
Connectors:	(Barrier strip)		
	2/56 screws 48 AI 12 AO 2 power	2/56 screws 48 DI/DO 2 + 5V power	6/32 screws 32 DI/DO 2 + 5V power
	(J1, J2, J3 to iSBC boards)		
	50-pin 0.1 in. centers (2.54 mm)	50-pin 0.1 in. centers (2.54 mm)	50-pin 0.1 in. centers (2.54 mm)
	(Mating connector: 3M 3415-0000 or TI H3-12125)		

Maximum Distance from iSBC® Boards

The iCS 910/920/930 panels are shipped with 4-ft. long cables. With customer provided 50-conductor or twisted pair ribbon cable, however, the iCS 910/920/930 panels can be mounted remote from the iSBC analog or digital I/O boards. In electrically quiet environments using normal iSBC board line driver/receivers, the iCS 910/920/930 panels should be able to operate up to 25 ft. (7.69m) from the iSBC board.

Electrical Characteristics

Power Requirements

iCS 920 panel — +5V ±5%, 1.46A max (24 channels high current drive)

ICS 920 Channel Configuration	Maximum per Channel Current (Includes pullups, LEDs, isolators, drivers)
TTL in	23 mA
TTL out	23 mA
Opto-isolated in	23 mA
Opto-isolated out	41 mA
Open collector driver output	61 mA

Note: Both iCS 920 and iCS 930 panels have jumpered provision for externally supplied +5V power via a screw terminal block.

iCS 930 panel — +5V ±5%, 320 mA max. Output AC or DC channel: 20 mA/chan max; Input AC or DC channel: 12 mA/chan max.

Maximum Power Dissipation

iCS 910 panels — 3 watts with 16 channels analog input signal conditioning

iCS 920 panels — 12 watts with 24 channels each containing high current driver outputs

iCS 930 panels — 80 watts with 16 channels of AC or DC output

Underwriters Laboratory (UL) Listing

The iCS 910/920/930 signal conditioning/termination panels are UL listed components under the UL safety standard for process control equipment, UL 1092.

Environmental Characteristics

Operating Temperature — 0 to 70°C (32°F to 158°F)

Relative Humidity — 0 to 90%, noncondensing

Hardware Supplied

iCS 910 — Analog Signal Conditioning/Terminating Panel, three 4-ft., 50-conductor flat ribbon cables with connectors, and installation kit.

iCS 920 — Digital Signal Conditioning/Termination Panel, one 4-ft., 50-conductor flat ribbon cable with connectors, and installation kit.

iCS 930 — AC Signal Conditioning/Termination panel, one 4-ft., 50-conductor ribbon cable with connectors, and installation kit.

Installation kit consists of RETMA (19″ rack) mounting bracket, clear plastic safety cover, labelling strip with blank gummed labels, hex standoffs and mounting screws.

Documentation Supplied

A schematic diagram and assembly diagram are supplied with each iCS 910/920/930 panel.

Reference Manuals

9800800A — iCS 910 Analog Signal Conditioning/Termination Panel Hardware Reference Manual (Order Separately)

9800801A — iCS 920 Digital Signal Conditioning/Termination Panel Hardware Reference Manual (Order Separately)

9800802A — iCS 930 AC Signal Conditioning/Termination Panel Hardware Reference Manual (Order Separately)

Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

Installation

Complete instructions for installation and service are contained in the applicable iCS 910/920/930 Hardware Reference Manual.

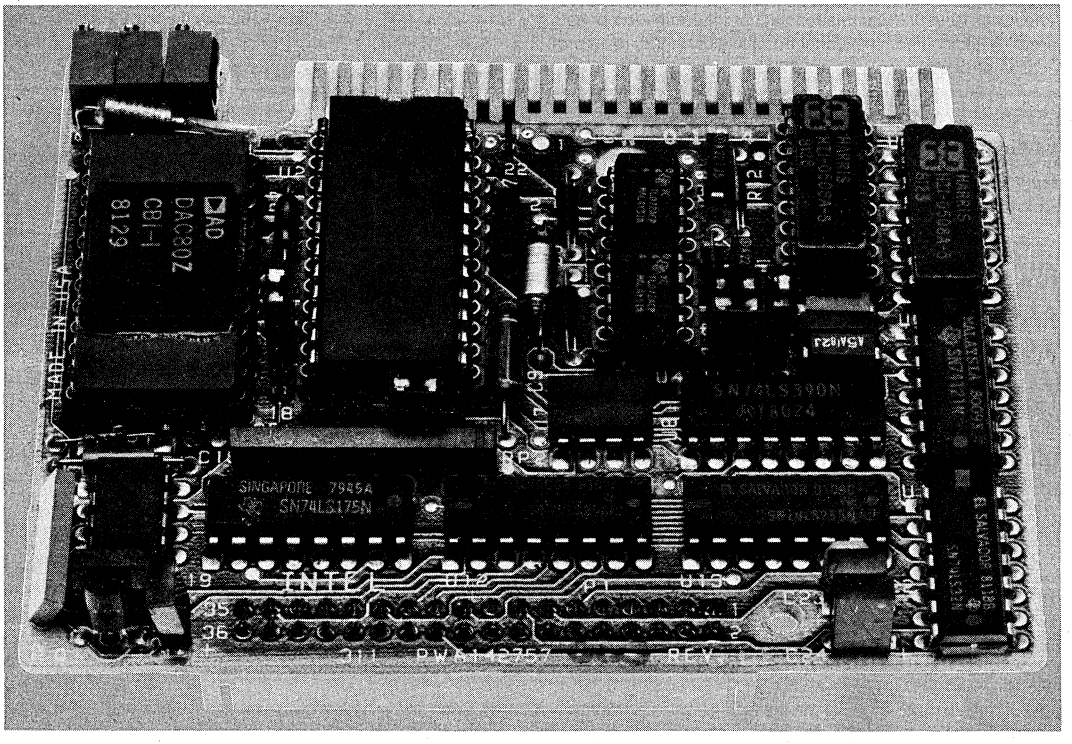
ORDERING INFORMATION

Part Number	Description
iCS 910	Analog signal conditioning/ termination panel
iCS 920	Digital signal conditioning/ termination panel
iCS 930	AC signal conditioning/termination panel

iSBX™ 311
ANALOG INPUT MULTIMODULE™ BOARD

- **Low cost analog input for iSBX™ MULTIMODULE™ compatible iSBC® boards**
- **8 differential/16 single-ended, fault protected inputs**
- **20 mV to 5V full scale input range, resistor gain selectable**
- **Unipolar (0 to +5V) or bipolar (−5V to +5V) input, jumper selectable**
- **12-bit resolution analog-to-digital converter**
- **0.035% full scale accuracy (11 bits) at 25°C**
- **18 kHz samples per second throughput to memory**
- **Connector compatible with iCS 910 Analog Termination Panel**

The Intel iSBX 311 Analog Input MULTIMODULE board provides simple interfacing of non-isolated analog signals to any iSBC board which has an iSBX compatible bus and connectors. The single-wide iSBX 311 plugs directly onto the iSBC board, providing data acquisition of analog signals from eight differential or sixteen single-ended voltage inputs, jumper selectable. The iSBX 311 MULTIMODULE is connector and pinout compatible with the Intel iCS 910 Analog Signal Conditioning/Termination panel so that field wiring can easily be terminated and current loop-to-voltage conversion resistors can be mounted for current loop analog signal monitoring. Resistor gain selection is provided for both low level (20mv full scale range) and high level (5 volt FSR) signals. Incorporating the latest high quality IC components, the iSBX 311 MULTIMODULE board provides 12 bit resolution, 11 bit accuracy, and a simple programming interface, all on a low cost iSBX MULTIMODULE board.



FUNCTIONAL DESCRIPTION

The iSBX 311 Analog Input MULTIMODULE board is a member of Intel's growing family of MULTIMODULE expansion boards, designed to allow quick, easy, and inexpensive expansion for the Intel single board computer product line. The iSBX 311 Analog Input MULTIMODULE Board shown in figure 1, is designed to plug onto any host iSBC microcomputer that contains an iSBX bus connector (P1). The board provides 8 differential or 16 single-ended analog input channels that may be jumper-selected as the application requires. The MULTIMODULE board includes a user-configurable gain, and a user-selectable voltage input range (0 to +5 volts, or −5 to +5 volts). The MULTIMODULE board receives all power and control signals through the iSBX bus connector to initiate channel selection, sample and hold operation, and analog-to-digital conversion.

Input Capacity

Sixteen separate analog signals may be randomly or sequentially sampled in single-ended mode with the sixteen input multiplexers and a common ground. For noisier environments, differential input mode can be configured to achieve 8 separate differential signal inputs, or 16 pseudo-differential inputs.

Resolution

The iSBX 311 MULTIMODULES provide 12-bit resolution with a successive approximation analog-to-digital converter. For bipolar operation (−5 to +5 volts) it provides 11 bits plus sign.

Speed

The A-to-D converter conversion speed is 35 microseconds (28KHZ samples per second). Combined with the sample and hold, settling times and the programming interface, maximum throughput via the iSBX bus and into memory will be 54 microseconds per sample, or 18 KHZ samples per second, for a single channel, a random channel, or a sequential channel scan. A-to-D conversion is initiated via the iSBX connector and programmed command from the iSBC base board. Interrupt on end-of-conversion is a standard feature to ease programming and timing constraints.

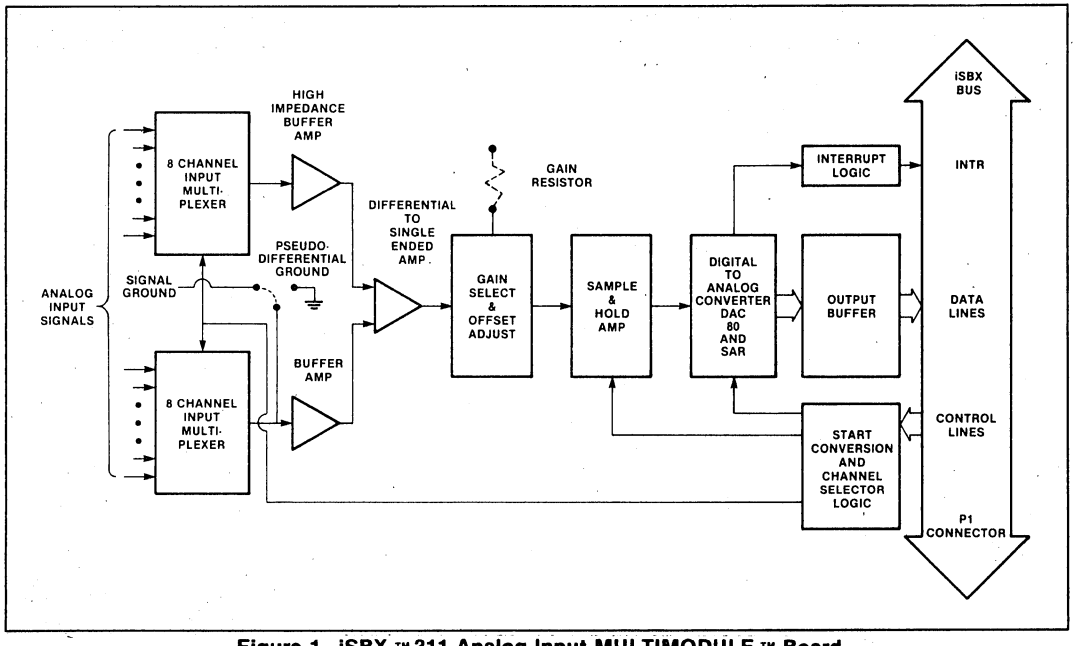


Figure 1. iSBX ™ 311 Analog Input MULTIMODULE ™ Board

AFN-01628A

Accuracy

High quality components are used to achieve 12 bits resolution and accuracy of .035% full scale range ± ½ LSB. Offset and gain are adjustable to ±0.024% FSR ± ½ LSB accuracy at any fixed temperature between 0°C (gain = 1). See specifications for other gain accuracies.

Gain

To allow sampling of millivolt level signals such as strain gauges and thermocouples, gain is made configurable via user inserted gain resistors up to 250× (20 millivolts, full scale input range). User can select any other gain range from 1 to 250 to match his application.

OPERATIONAL DESCRIPTION

The host iSBC microcomputer addresses the iSBX 311 MULTIMODULE board by executing IN or OUT instructions to the iSBX 311 MULTIMODULE as one of the legal port addresses. Analog-to-digital conversions can be programmed in either of two modes: 1. start conversion and poll for end-of-conversion (EOC), or 2. start conversion and wait for interrupt (INTRO/) at end of conversion. When conversion is complete as signaled by one of the above techniques, INput instructions read two bytes (low and high bytes) containing the 12 bit data word plus status information as shown below.

OUTput Command — Select input channel and start conversion.

Bit Position	7	6	5	4	3	2	1	0
Input Channel					C3	C2	C1	C0

INput Data — Read converted data and status (low byte) or Read converted data (high byte). Reads can be with or without reset of interrupt request line (INTRO/).

Bit Position	7	6	5	4	3	2	1	0
Low/status Byte	D3	D2	D1	D0		start/	busy/	EOC/

High Byte	D11	D10	D9	D8	D7	D6	D5	D4

Fastest data conversion and transfer to memory can be obtained by dedicating the microcomputer to setting the channel address/starting conversion, polling the status byte for EOC/, and when it comes true, read the two bytes of the conversion and send the start conversion/next channel address command. For multitasking situations it may be more convenient to use the interrupt mode, reading in data only after an interrupt signals end of conversion.

SPECIFICATIONS

Inputs — 8 differential. 16 single-ended. Jumper selectable.

Full Scale Input Voltage Range — −5 to +5 volts (bipolar). 0 to +5 volts (unipolar). Jumper selectable.

Gain — User-configurable through installation of two resistors. Factory-configured for gain of X1; gains above 250 not recommended.

Resolution — 12 bits over full scale range (1.22 mv at 0-5 v, 5 μv at 0-20 mv)

Accuracy —

Gain	Accuracy at 25°C
1	±0.035% ± ½ LSB
5	±0.035% ± ½ LSB
50	±0.035% ± ½ LSB
250	±0.035% ± ½ LSB

NOTE:
Figures are in percent of full scale reading. At any fixed temperature between 0° and 60°C, the accuracy is adjustable to ±0.035% of full scale.

Dynamic Error — ±0.015% FSR for transitions

Gain TC (at Gain = 1): 30 PPM per degree centigrade (typical); 56 PPM per degree centigrade (max).

Offset TC (in percent of FSR/°C):

Gain	Offset
1	.0018
5	.0036
50	.024
250	.116

Offset is measured with user-supplied 10 PPM/°C gain resistors installed.

Input Protection — ±30 volts.

Input Impedance — 20 megohms (minimum).

Conversion Speed — 50 microseconds (nominal).

Common Mode Rejection Ratio — 60 db (minimum).

Sample and hold — sample time 15 microseconds.

Aperature — hold aperature time: 120 nanoseconds.

Connectors —

Interface	Pins (Qty)	Centers in	Centers cm	Mating Connectors
P1 iSBX Bus	36	0.1	0.254	iSBC iSBX connector
J1 8/16 channels analog	50	0.1	0.254	3m 3415-000 or T1 H312125 or iCS 910 cable

Physical Characteristics

Width — 9.40 cm (3.7 inches)

Length — 6.35 cm (2.5 inches)

Height — 2.03 cm (0.80 inch) MULTIMODULE board only

2.82 cm (1.13 inches) MULTIMODULE and iSBC board

Weight — 68.05 gm (2.4 ounces)

Electrical Characteristics (from iSBX connector)

Vcc = ±5 volts (±0.25V), Icc = 250 mAmax

Vdd = +12 volts (±0.6V), Idd = 50 mAmax

Vss = −12 volts (±0.6V), Iss = 55 mAmax

Environmental Characteristics

Operating Temperature — 0° to 60°C (32° to 140°C)

Relative Humidity — to 90% (without condensation)

Reference Manuals

142913-001 — iSBX 311 Analog Input MULTIMODULE Board Hardware Reference Manual (order separately)

Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

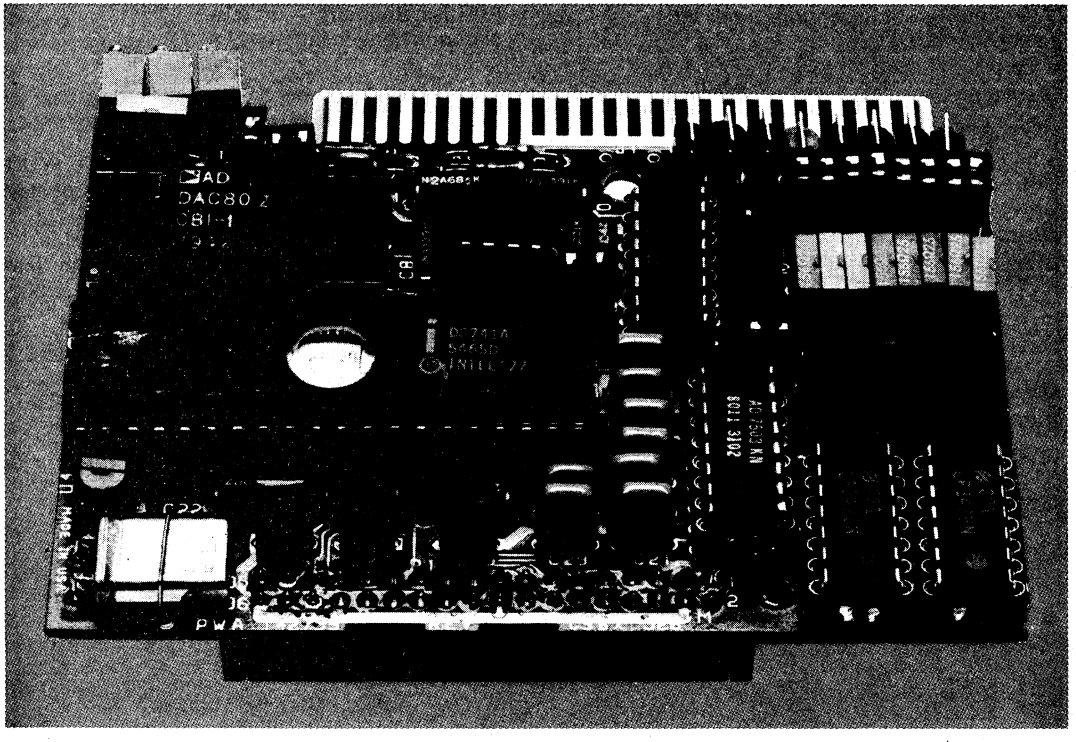
ORDERING INFORMATION

Part Number	Description
SBX 311	Analog Input MULTIMODULE Board

iSBX™ 328
ANALOG OUTPUT MULTIMODULE™
EXPANSION BOARD

- **Low cost analog output for iSBX™ MULTIMODULE™ compatible iSBC® Boards**
- **8 channels output, current loop or voltage in any mix**
- **4-20 mA current loop; 5V unipolar or bipolar voltage output**
- **12-bit resolution**
- **0.035% full scale volage accuracy @ 25°C**
- **Connector compatible with ICS 910 Analog Termination Panel**
- **Intel design based on UPI control for high density and low cost**
- **Programmable offset adjust in current loop mode**

The Intel iSBX 328 MULTIMODULE board provides analog signal output for any iSBC board which has an iSBX compatible bus and connectors. The single-wide iSBX 328 plugs directly onto the iSBC board, providing eight independent output channels of analog voltage for meters, CRT control, programmable power supplies, etc. Voltage output can be mixed with current loop output for control of popular 4-20ma industrial control elements. By using an Intel single chip computer LSI (8041) for refreshing separate sample-hold amplifiers through a single 12 bit DAC, eight channels can be contained on a single MULTIMODULE board for high density and low cost per channel. High quality analog components provide 12 bit resolution, 11 bit accuracy, and slew rates per channel of 0.1 volt per microsecond. Programming the iSBX 328 MULTIMODULE board is done via a simple two byte protocol over the iSBX bus. Maximum channel update rates are 5KHZ on a single channel to 1 KHZ on all eight channels. Outputs are compatible for screw termination of field wiring on the ICS 910 Analog Signal Conditioning/Termination Panel.



FUNCTIONAL DESCRIPTION

The iSBX 328 MULTIMODULE board, shown in figure 1 is designed to plug onto any host iSBC microcomputer that contains an iSBX bus connector. The board uses an 8041 UPI device to control eight analog output channels that may be user-configured through jumpers to operate in either bipolar voltage output mode (−5 to +5 volts), unipolar voltage output mode (0 to +5 volts), or current loop output mode (4 to 20 mA) applications. Channels may be individually wired for simultaneous operation in both current loop output and voltage output applications. The outputs from 50-pin edge connector J1 on the MULTIMODULE board are pin-compatible with the iCS 910 Signal Conditioning/Termination Panel.

Interfacing Through the Intel iSBX Bus

All data to be output through the MULTIMODULE board is transferred from the host iSBC microcomputer to the MULTIMODULE board via the iSBX bus connector. The UPI device on the MULTIMODULE board accepts the binary digital data and generates a 12-bit data word for the Digital-to-Analog Converter (DAC) and a four bit channel decode/enable for selecting the output channel. The DAC transforms the data into analog signal outputs for either voltage output mode or current loop output mode. Offsetting of the DAC voltage in current output mode may be performed by the UPI software offset routine or by the hardware offset adjustments included on the board. The MULTIMODULE board status is available via the iSBX bus connector, to determine if the UPI is ready to receive updates to analog output channels.

OPERATIONAL DESCRIPTION

The host iSBC microcomputer addresses the MULTIMODULE board by executing IN or OUT instructions specifying the iSBX 328 MULTIMODULE as a port address. The UPI on the iSBX 328 is initialized to select whether software or hardware offset is to be used and how many channels will be active. Then a 2 byte transfer to each active channel sets the 12 bit output value, the channel selected and the current or voltage mode.

Commands

OUTput Command — Initialization of UPI/iSBX 328

7							0 Bit
			N	N	D2	D1	D0

last channel to be output

NN: 0,0 = unipolar configuration software current offset
0,1 = no mixing
1,0 = bipolar configuration software current offset

OUTput Command — Data Bytes

	7							0 Bit
High Byte	D11	D10	D9	D8	D7	D6	D5	D4
Low Byte	D3	D2	D1	D0	A3	A2	A1	V/C

DAC Data — DAC channel to receive data

0 = UPI generates offset
1 = SBC generates offset in current loop mode

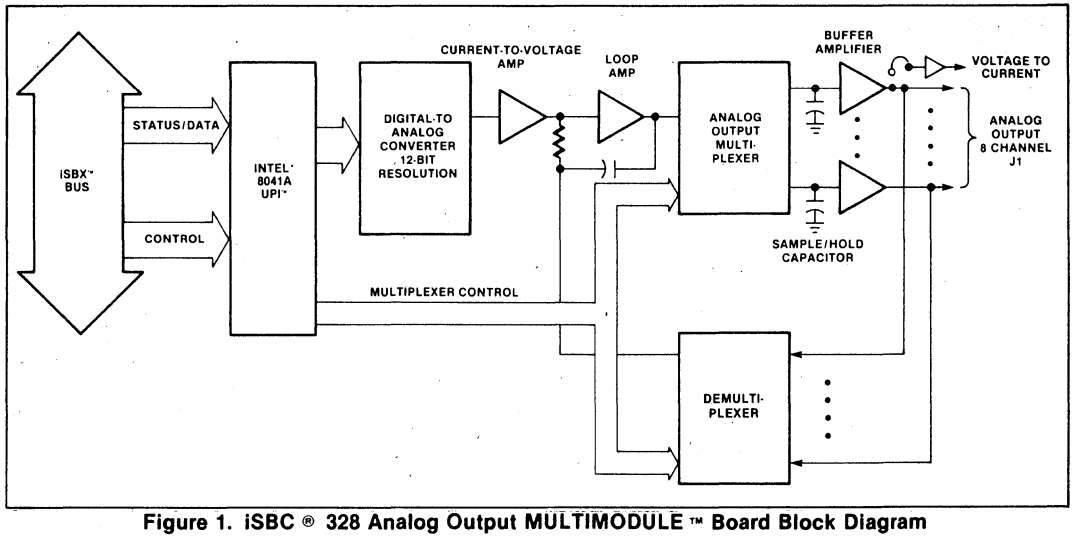
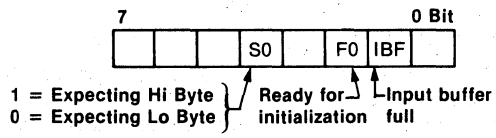


Figure 1. iSBC ® 328 Analog Output MULTIMODULE ™ Board Block Diagram

AFN-01627A

INput Command — Status Buffer Read



Interrupts

No interrupts are issued from the iSBX 328 to the host iSBC microcomputer. Data coordination is handled via iSBC software polls of the status buffer.

SPECIFICATIONS

Outputs — 8 non-isolated channels, each independently jumpered for voltage output or current loop output mode.

Voltage Ranges — 0 to +5 volts (unipolar operation)
−5 to +5 volts (bipolar operation)

Current Loop Range — 4 to 20 mA (unipolar operation only)

Output Current — ±5 mA maximum (voltage mode-bipolar operation)

Load Resistance — 0 to 250 ohms with on-board iSBX power. 1000 ohms minimum with 30 VDC max. external supply

Compliance Voltage — 12 V using on-board iSBX power. If supplied by user, up to 30 VDC max

Resolution — 12 bits bipolar or unipolar

Slew Rate — 0.1 volt per microsecond minimum

Single Channel Update Rate — 5KHz

Eight Channel Update Rate — 1KHz

Accuracy —

Mode	Accuracy	Ambient Temp
Voltage-Unipolar, typical	±0.025% FSR	@ 25°C
Voltage-Unipolar, maximum	±0.035% FSR	@ 25°C
Voltage-Unipolar, typical	±0.08% FSR	@ 0° to 60°C
Voltage-Unipolar, maximum	±0.19% FSR	@ 0° to 60°C
Voltage-Bipolar, typical	±0.025% FSR	@ 25°C
Voltage-Bipolar, maximum	±0.035% FSR	@ 25°C
Voltage-Bipolar, typical	±0.09% FSR	@ 0° to 60°C
Voltage-Bipolar, maximum	±0.17% FSR	@ 0° to 60°C
Current Loop, typical	±0.07% FSR	@ 25°C
Current Loop, maximum	±0.08% FSR	@ 25°C
Current Loop, typical	±0.17% FSR	@ 0° to 60°C
Current Loop, maximum	±0.37% FSR	@ 0° to 60°C

Refresh and Throughput Rates**	
Refresh 1 channel (no new data):	80 us
Refresh all 8 channels (no new data):	650 us
Update and refresh 1 channel with new data: firmware program 2	150 us
for each additional channel	130 us
Update and refresh 1 channel with new data: firmware program 1 or 3	200 us
for each additional channel	155 us
Update and refresh all 8 channels (all new data): firmware program 2	1.050 ms
per channel of new data	50 us
Update and refresh all 8 channels (all new data): firmware program 1 or 3	1.280 ms
per channel of new data	80 us

** All times nominal

Output Impedance — 0.1 ohm. Drives capacitive loads up to 0.05 microfarads. (approx. 1000 foot cable)

Temperature Coefficient — 0.005%/°C

Connectors —

Interface	Pins (Qty)	Centers in	Centers cm	Mating Connectors
P1 iSBX Bus	36	0.1	0.254	iSBC iSBX connector
J1 8/16 channels analog	50	0.1	0.254	3m 3415-000 or T1 H312125 or iCS 910 cable

Physical Characteristics

Width — 9.40 cm (3.7 inches)

Length — 6.35 cm (2.5 inches)

Height — 1.4 cm (0.56 inch) MULTIMODULE board only
2.82 cm (1.13 inches) MULTIMODULE and iSBC board.

Weight — 85.06 gm (3.0 ounces)

Electrical Characteristics

Vcc = ±5 volts (±0.25V), Icc = 140 ma max

Vdd = ±12 volts (±0.6V), Idd = 45 ma max
(voltage mode)
= 200 ma max
(current loop
mode)

Vss = −12 volts (±0.6V), Iss = 55 ma max

Environmental Characteristics

Operating Temperature — 0° to 60°C (32° to 140°C)

Relative Humidity — to 90% (without condensation)

Reference Manuals

142914-002 — Input Power — iSBX 328 Analog Output MULTI-MODULE Board Hardware Reference Manual (Order Separately)

Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051

ORDERING INFORMATION

Part Number	Description
SBX 328	Analog Output MULTIMODULE Board

intel®

APPLICATION NOTE

AP-52

March 1979

Using Intel's Industrial Control Series in Control Applications

Peter Andersen
OEM Microcomputer Systems Applications

I. INTRODUCTION

The introduction of the single board computer as a tool for the system designer has opened the way for many varied application areas to benefit from the advantages of computer utilization. A problem still exists, however, because the available I/O configurations have been largely incompatible with the wiring and packaging techniques required in industrial environments. This problem is overcome by the utilization of the Intel® iCS™ product family. The purpose of this application note is to provide a representative approach to the implementation of a computerized solution to an industrial control system.

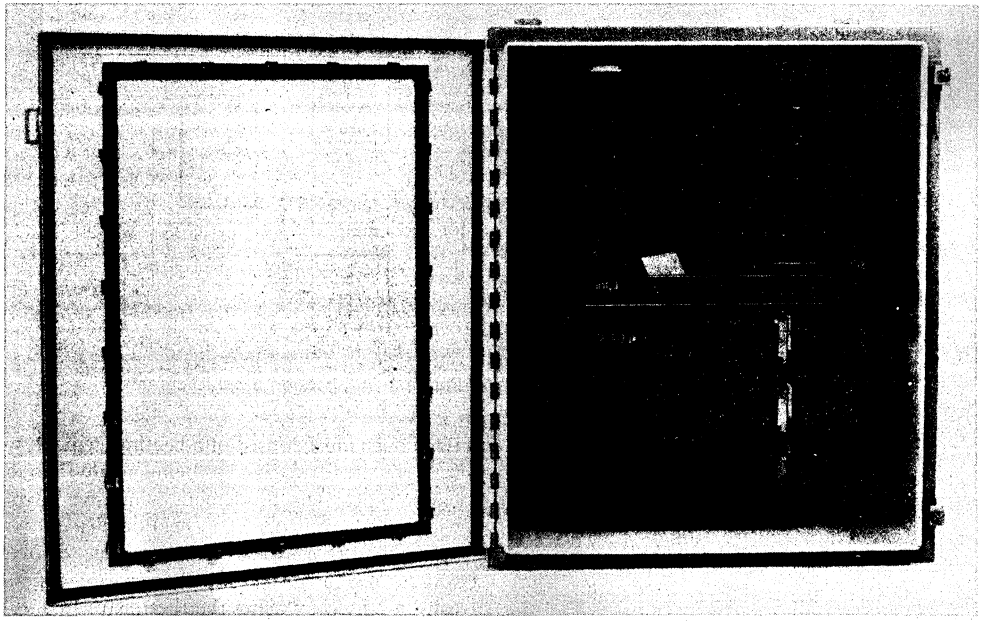
System Description

This application note will deal with a control system which will regulate the temperature in each of four ovens. Each oven will be defined as utilizing a light bulb for heating. Normal convection will be used to provide cooling. The internal temperature will be measured by means of a thermistor installed in each oven. We will assume that we will be required to implement some type of operator panel near the ovens which will allow the status of each oven to be monitored. This approach is similar to many common industrial applications which require a supervisory control station in one area and a separate operator interaction panel near the equipment being controlled. The setpoint and tolerances should be input from an external location.

With these facts about our system defined, we can begin a step by step solution to providing a computerized control system to operate the ovens. We will discuss the various equipment trade-offs and the decisions which will be used to define the hardware/software designs.

Control Algorithm

Before we can begin the design of our system, we must have a clear idea of the technique we will use to control the system. Our control system must maintain the oven temperature within a predefined and fairly narrow range of the setpoint. Let us make an assumption that the light bulb will be controlled digitally, meaning that the bulb must either be turned fully on or it must be turned fully off. The obvious control technique then becomes turning the bulb on when the temperature of the oven is below our lower limit and turning the bulb off when the temperature is above the higher limit. It seems reasonable to assume that this technique will provide a temperature in the oven which varies sinusoidally with time. This is true because even though the lamp is turned off, it will continue to generate heat for a short period of time. Likewise, when the bulb is turned on, it will not instantly be able to provide heat to raise the temperature of the



chamber. We would expect to have a system response such as is shown in Figure 1. A better method of control can be devised if we provide some type of temperature prediction into our control algorithm. Since this utilizes the rate of temperature increase or decrease, it will involve a type of derivative control system. This derivative control action will tend to dampen the temperature oscillations which might be encountered if only an instantaneous on-off control system were utilized. Figure 2 shows the response with time that we might expect with this type of control system.

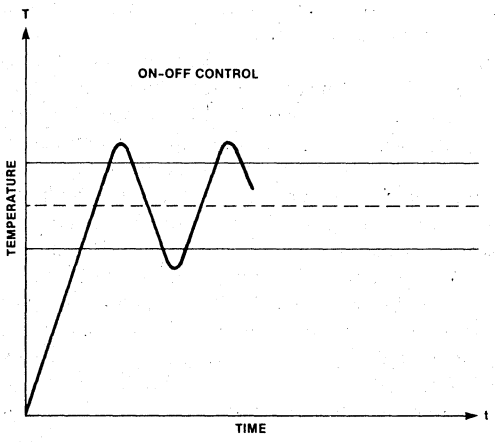


Figure 1. Maximum Effort Current Temperature

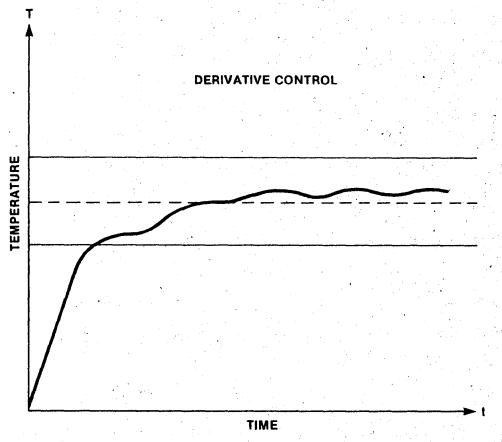


Figure 2. Maximum Effort Projected Temperature

The second approach is superior to the first because the control will provide a much smaller oscillation of the oven temperature. Other solutions are possible, such as providing a modulated output to the lamp. However, in an attempt to provide a simple model upon which to expand our system solution, we will assume that the second approach will provide us with an accurate enough control of the oven temperature.

Having made the decision as to the control technique, we can proceed with the task of determining the general system configuration. That is, we can define the physical system characteristics and the components to which we must interface the computer system. This approach is identical to that which would be used in a conventional control system design.

Basic System Configuration

Based upon the data which we have provided so far, it is possible to build a block diagram of the system's major components. The system consists of four ovens, an operator's panel, a data entry panel, and the actual control logic. A block diagram for the system is shown in Figure 3. We must now further define the elements which make up each of these blocks.

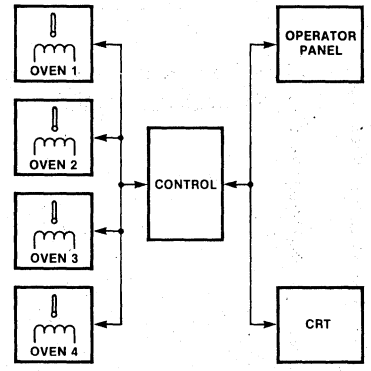


Figure 3. Application Block Diagram

Each oven must consist of a heating element, which we have already defined as being a light bulb, and a temperature sensing element which we have said will be a thermistor. Each heating element will be switched on or off by applying or removing a source of 115 VAC. The thermistor temperature can be sensed by using the thermistor in a voltage

divider circuit. We can then measure the voltage across a fixed resistor to obtain an analog signal which is proportional to the oven temperature. We will determine the required value of the fixed resistor at a later time.

The operator's panel should be designed to provide the workfloor operator with basic information as to the status of each oven. It should also allow some method by which he can inhibit the operation of any oven should it become necessary for charging or servicing the oven. We can then define the basic elements which should make up the operator's control panel. Each oven should have associated with it the following controls and indicators:

1. Oven ON/OFF Switch — This switch will allow the operator to inhibit the oven operation by turning the appropriate oven switch to OFF.
2. Oven RUNNING Indicator — This indicator will provide a visual indication that the oven is activated and that the temperature is being controlled.
3. Oven IN TOLERANCE Indicator — This indicator will turn on when the oven temperature falls within the allowable bandwidth around the setpoint for that oven.
4. Oven ALARM Indicator — This indicator is the complement of the in tolerance lamp. It will be turned on when the oven is activated and the temperature does not lie within the desired bandwidth.
5. Oven CAUTION Indicator — It may be necessary to alert the operator to a potential oven temperature control problem before it actually occurs and sets off the alarm indicator. Since we have defined our control algorithm as utilizing a type of derivative control, we can project the oven temperature ahead in time. We will turn the oven caution indicator on when we predict that the oven temperature will lie outside of the desired bandwidth in a predetermined future time period.

We have now defined the operator interface which we will utilize to control and monitor the oven processes.

At this point, we will make a decision that the interface used to input the setpoints will utilize a CRT terminal. Though the decision may seem to be completely arbitrary, we will see later that CRT terminals provide an extremely useful device for allowing an operator to communicate with the system. Once the decision has been made, we have no further requirements to consider hardware design for this terminal, as the entire operation can be handled in the software development which will be considered later.

A common technique for documenting a system is the ladder diagram. At this time, we can construct a ladder for our control system. Unlike conventional design techniques, our ladder diagram need only be concerned with the actual drive and sensing circuits since the logic required to drive the various outputs will be defined using software. This results in a considerable simplification of the design process. A ladder diagram for a typical oven is shown in Figure 4. We can defer the implementation of the control algorithm until we begin to develop the software portion of our control system. It is now possible to complete the external hardware design and to implement the system wiring package.

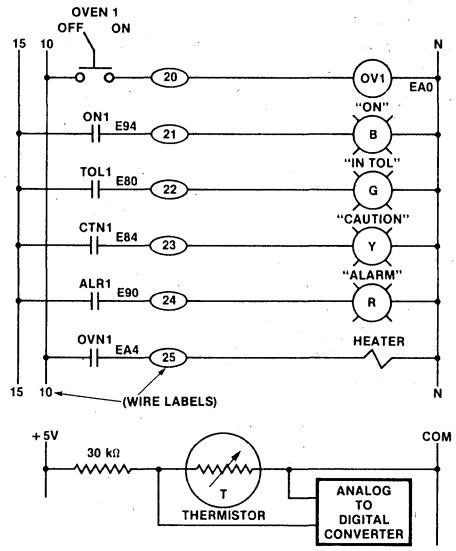


Figure 4. Ladder Diagram of One Oven

II. WIRING INTERFACES

A major pitfall in utilizing a computer for control systems has traditionally been the requirement for the design engineer to expend a considerable amount of his time in designing interfaces to connect the physical wiring to the computer system. The introduction of Intel's product line of termination panels has essentially eliminated the require-

ment of designing interfaces and allows more engineering time to be spent providing a solution to the application. Before we continue with the specific design, we should spend some time discussing the various types of termination panels available and the general characteristics of each panel.

Analog Termination Panels

The Intel® iCS 910 Analog Termination Panel has been designed to provide a simple means of terminating the analog wiring and of providing an interface to the control system input/output. All wiring is terminated utilizing pressure type screw barrier blocks. Termination blocks have been provided to allow the termination of up to 32 single-ended or 16 differential channels of analog input. For use in a differential input environment, such as we will be using, the terminator blocks provide wiring terminations compatible with shielded cable inputs in that provision has been made to accept the shield of each input signal. The shield is then carried through the on-board circuits to the analog-to-digital converter. Provision has been made on the board for the mounting of commonly used circuits for signal conditioning. The available signal conditioning circuits provide for installation of current termination resistors and the installation of a single pole low pass filter network. The basic barrier assignments for the iCS 910 termination panel are shown in Figure 5. The possible circuit networks for this panel are illustrated in Figure 6. A complete description of the analog termination panel can be found in the *iCS 910 Analog Signal Conditioning/Termination Panel Hardware Reference Manual* (manual order number 9800800A).

The functions of the analog termination panel will become more clear as we develop the actual configuration required to support our oven application. Referring to the ladder diagram (Figure 4) we see that a fixed resistor is necessary to provide the voltage divider network to sense the oven temperature. The current termination resistor (Rc) on the iCS 910 board can be used to provide a convenient mounting location for this component (refer to iCS 910 circuit schematic, Figure 6). At this point, we must make a design decision regarding the utilization of a low pass filter for our analog circuits. Since the oven temperatures are not expected to exhibit rapid fluctuations with time, the use of a low pass filter will not adversely effect the temperature

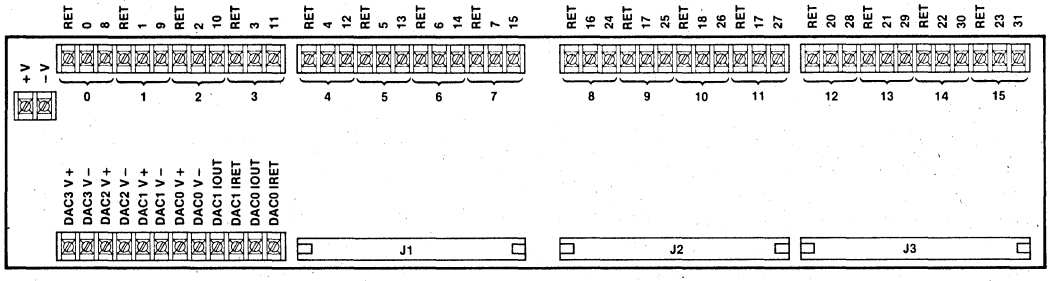


Figure 5. iCS™ 910 Analog Terminator Panel Assignments

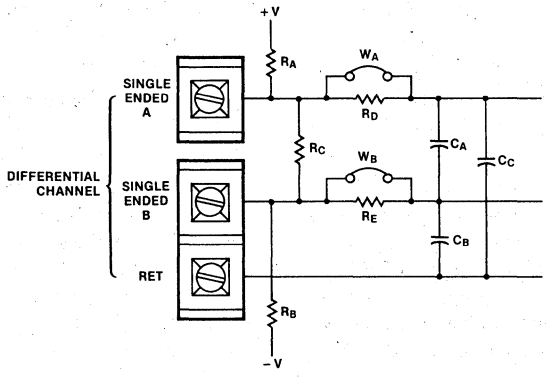


Figure 6. Typical Circuit on Analog Terminator

sensing. Indeed, the use of a low pass filter should contribute to spurious signal rejection should the analog cables pick up external noise signals. Calculations will show that the use of a filter network consisting of 11K ohms series resistors and a 2.2μF capacitor will provide the filter characteristics shown in Figure 7.

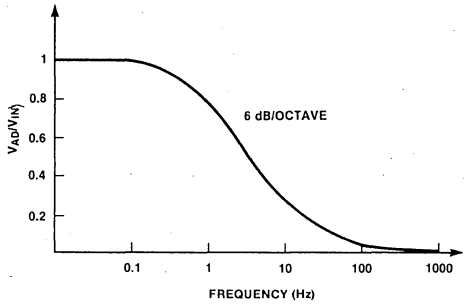


Figure 7. Single Pole Filter Characteristics

Based upon our requirements and using the circuit schematic of Figure 6, we can provide the circuit interfaces required by our ladder diagram (Figure 4) by configuring the channels of the iCS 910 terminator as shown in Figure 8. This results in a simple two-wire per oven analog interface. The terminator board is designed to connect to the various analog I/O boards by means of a standard ribbon cable which is supplied with the terminator panel. The actual selection of the appropriate analog board will be deferred until later. We will define that oven number 1 will correspond to the differential analog channel 0; oven will correspond with channel 1; oven 3 will correspond with channel 2; and oven 4 will use channel 3. This leaves 12 analog differential channels available for future expansion. The channel selection just made was a purely arbitrary choice.

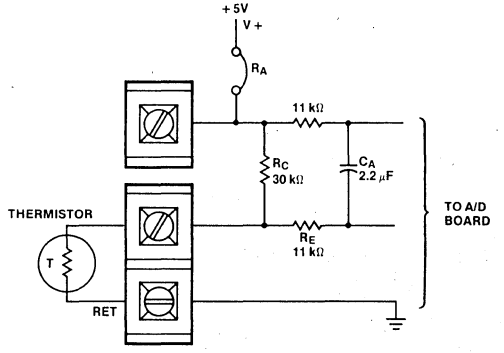


Figure 8. Analog Circuit for Oven Application

The wiring to the iCS 910 terminator panel can then be made essentially as shown in Figure 9. Clearly, the use of the terminator panel greatly simplified the connection between the control sys-

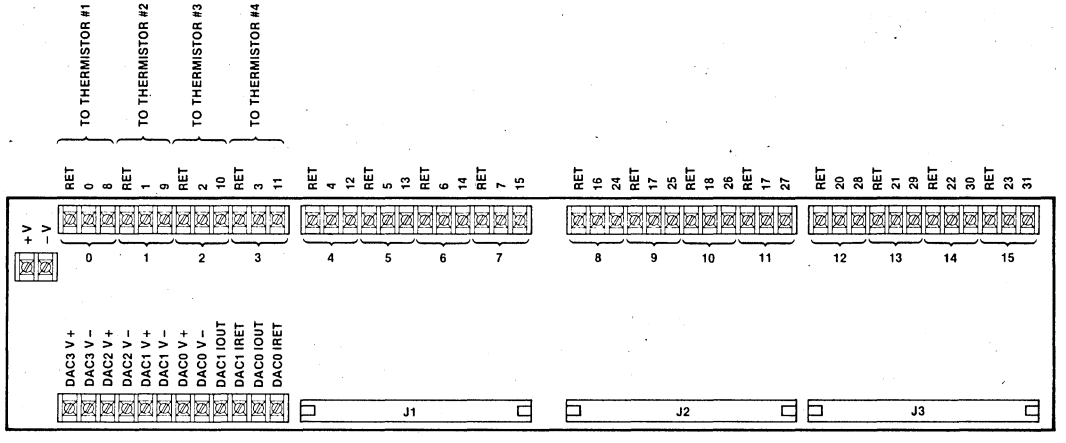


Figure 9. Analog Terminator Wiring

tem and the physical devices which are to be monitored or controlled. Figure 10 shows the placement of the components onto the board.

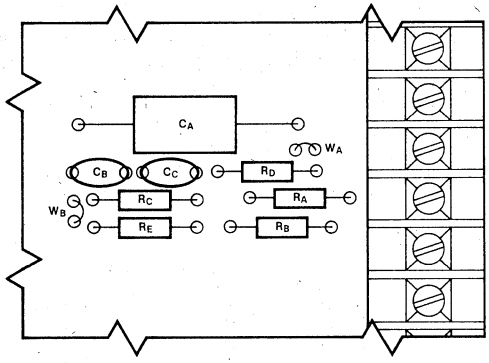


Figure 10. Analog Terminator Component Locations

Low Voltage Digital Termination Panels

Looking again at our ladder diagram for an oven control system (Figure 4), we see the need to provide a second type of interface signal. This is to provide the switching for the various indicator lamps used on the operator's control panel. Traditionally, this interface has been handled by using electromechanical relays. The coils would be driven by the low voltage control system and the relay contacts were used to drive the external indicators. Modern technology provides us with a solid state device to perform the same function, the optical isolator. We can use these devices to provide a highly reliably and low cost alternative to the relay interface. The Intel® iCS 920™ Digital Signal Conditioning/Terminator Panel provides us with a convenient vehicle for mounting the optical isolator circuits and for terminating the wiring associated with the indicator devices.

The iCS 920 panel is designed to be used by those interface circuits which incorporate operating voltages less than 50 volts and which generally use currents which are smaller than 300 mA. These limits are given only for a general guideline since a wide variety of optical isolators and drivers are available for use on the board. Some of the devices are capable of handling greater voltages or currents. A representative list of available devices and complete details of the termination panel are available in the *iCS 920 Digital Signal Conditioning/Termination Panel Hardware Reference Manual* (manual order number 9800801A).

The digital panel provides terminations for up to 24 digital channels, each of which can be configured as either an input or an output channel according to the specific application requirements. As with the analog termination panel, all wire terminations are made using pressure type barrier strips which will accept up to 16 gauge wire. The 24 digital channels correspond with those input/output channels assigned to the standard Intel I/O configurations used on the single board computers and I/O expansion boards. We will dwell more on this subject later when we define the addresses associated with each circuit which we desire to incorporate into the termination panel.

Since the digital channels can be configured into either an input or an output mode, it is wise to discuss each configuration so that a clear understanding of the board can be obtained, even though our application example will only use the output mode with this board.

Figure 11 provides a schematic of the panel when it is configured for a digital input mode. To set up a channel to operate as an input, it is necessary to add at least two jumpers to the wire-wrap jumper posts. As can be seen, pins 6 and 4 must be connected together as well as pins 3 and 5. If the board is to provide a visual LED indication of the channel status, an additional jumper should be installed between pins 1 and 2 of the jumper posts. If this is done, be certain to take into account the additional current requirements when calculating the required input resistors. Two resistor mounting locations are provided to allow installation of selected components to handle the current limit through the optical isolator (Rx) and the threshold voltage for turn-on of the device (Ry). A complete and detailed procedure for selecting these resistors based upon the input voltages is provided in the iCS 920 hardware reference manual mentioned earlier. Provision has also been made on the termination panel for the installation of a diode (CR) to protect against reverse bias application.

The components have been placed on the board arranged in groups of two channels. This eases the task of finding various components or of locating the holes for installing the required components. This layout is illustrated in Figure 12. It is important to take note of the physical placement of the optical isolator chips in the 20-pin socket. This installation location must be followed rigorously when using a channel in an input mode. Also take note that provisions are provided for mounting two sizes of resistors in location (Rx). This will accom-

modate the power dissipation requirements which will be encountered in various application situations. Referring again to Figure 12, note that the **upper half of the layout represents odd channels and the lower portion of the layout is used for even channel component mounting.**

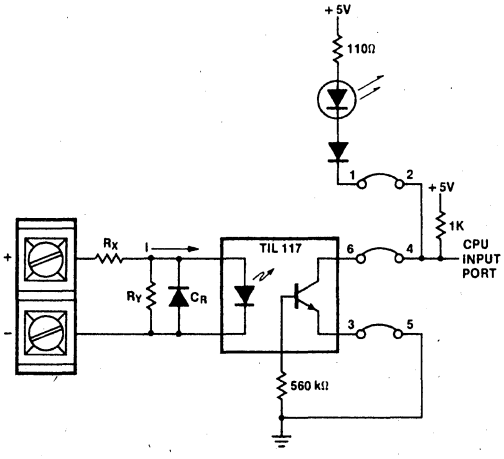


Figure 11. iCS™ 920 Digital Terminator Input Configuration

When the iCS 920 panel is used in this input mode, it corresponds to the utilization of a relay coil to sense some external contact closure. The resistors can be thought of as selecting the coil's operating voltage and the diode provides the same transient protection function as when installed on an electromechanical relay. Finally, the optical isolator output corresponds to the contacts associated with the relay coil. **As we will see later, this approach provides us with an unlimited number of contacts per relay coil.**

The oven application requires a contact for driving the indicator lamps associated with each oven. If we define the driving voltage to be 24 volts DC, we will find that the voltage and current requirements fall within the limits specified for using the iCS 920 Digital Signal Conditioning/Termination Panel. Let us examine in more detail how this can be accomplished.

We will select an industrial indicator assembly which utilizes a full voltage 24-volt lamp. Typical lamps would be type 387. This will require a drive of 40 mA at 28 volts. Our switching device must be capable of driving this load. The analogy used earlier to compare the optical isolator with a relay in an input mode holds true when we utilize the devices in an output configuration. If we examine the data sheet for the current switching characteristics of a typical optical isolator, say the TIL 113 (Appendix A), we can see that the current and voltage requirements fall well within the allowable ratings of the device. We have selected the relay contact characteristics! We need not concern ourselves with the selection of current limitation resistors (coil voltage ratings) since this circuitry is provided on the terminator panel when a circuit is

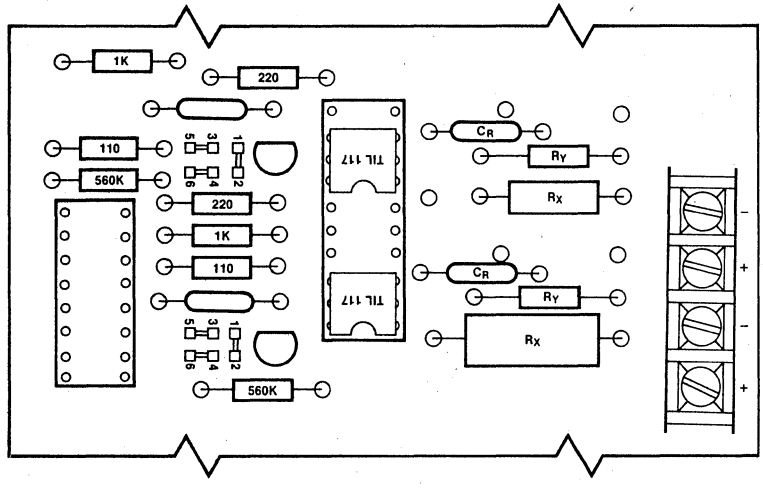


Figure 12. Digital Terminator Input Parts Layout

configured in an output mode. If we refer to Figure 13, we can see the on-board schematic for the output drive mode of operation. Two jumpers must be installed for each output channel. The first, between pins 1 and 2, is used to enable the LED channel status indicator. The second, between pins 3 and 4, actually connects the computer generated drive signal to the input of the optical isolator (analogous to connecting the relay coil to the driving line). Provision has been made on the circuit board for only one optional component in the output mode; this is the resistor (Rz). This component has the effect of increasing the response time of the switching device. Because our indicator lamps are not time critical, we will choose to omit the installation of this component.

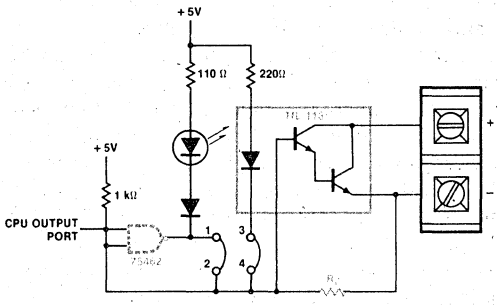


Figure 13. iCS™ 920 Digital Terminator Output Circuit

Figure 14 provides a drawing showing the location of the components on the iCS 920 panel when it is utilized as an output switch. Again note the placement of the optical isolators in the 20-pin sockets. Also note the jumper arrangement used to provide the required output circuitry.

Again referring to Figure 13, we see that an alternative to using the optical isolator for a switch exists. Provision has been made on the panel for the installation of high power buffer/driver chips such as the TI 75462. This device provides the same coil/contact characteristics as our optical isolator; however, no isolation between the input and output is provided. In certain applications, this configuration may be desirable and can be implemented by connecting jumpers 1 and 3 together, then placing a jumper block in the isolator socket location. The oven application will not use this mode because of the many advantages which isolation can provide.

Prior to actually installing the components onto the iCS 920 panel, it is necessary to assign the lamps to definite channel addresses. This involves making some additional assumptions and design configuration decisions. If we consider the total number of digital inputs and outputs which are required to handle all four ovens (including the as yet unconsidered switch and heater signals), we see that a total of 24 channels will be required. These will be broken out as shown below:

No. of Channels	Type	Function
16	DC	Oven indicator lamps
4	AC	Oven heaters
4	AC	Oven RUN switches

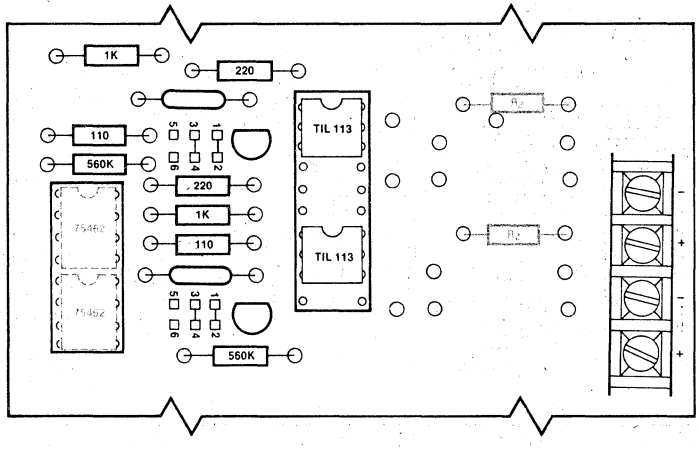


Figure 14. Digital Terminator Output Configuration

We have indicated that the 16 indicator lamps can be handled using the iCS 920 panel. An examination of the data sheets for the various Intel single board computers and expansion boards provides us with the fact that a common characteristic of most boards is the use of at least one Intel 8255 Programmable Peripheral Interface. This provides us with at least 24 I/O lines with which to work on each single board computer. We can then assume that we will not require an I/O expansion board to implement our application. Ideally, we can handle our total requirements with one parallel interface.

The various Intel parallel ports are brought off of the computer and expansion boards using edge connectors. These edge connectors are then connected to the termination panels using a standrd ribbon cable assembly, effectively providing an extension of the I/O ports out to the termination panels. The 24 channels are grouped into three I/O ports (each consisting of 8 channels or bits) which are then called port A, port B, and port C. When connected to the iCS 920 panel, these ports and their bit assignments will be as shown in Figure 15.

At this point, we seem to be in a dilemma since we would like to use all 24 channels and we have used only 16 of them on our panel while we have utilized the edge connector of the interface. It would be desirable to have some technique to extend the other 8 channels to a high voltage terminator panel. It might be well to interrupt our channel assignments at this time to jump ahead and consider the features of the iCS product line which will enable us to accomplish our interface desires. We will then consider the interface of the high voltage signals to our control system before returning to the problem of assigning port locations to our lines.

High Voltage Digital Termination Panels

The Intel® iCS 930™ AC Signal Conditioning/ Termination Panel is designed to interface up to 16 AC signals (up to 280 volts at 3 amps) or high current DC signals (up to 50 volts at 3 amps) to the parallel ports of the Intel single board computers or I/O expansion modules. The barrier strip terminations on this panel are designed to easily handle the 14 gauge wire commonly found in applications requiring the use of the AC terminator.

Solid state relays are used to provide the interface between the computer I/O ports and the physical plant devices. These devices make the utilization of the panel a simple task once a ladder diagram of the required circuits has been drawn. As we have previously mentioned and as is clear from looking at Figure 4, we shall need to utilize eight of the available circuits, four for input and four for output. The implementation of each signal type requires only that we insert the correct type of solid state relay into the appropriate socket.

First, consider the input configuration which is required to sense the position of the oven RUN switches. Figure 16 shows the circuit schematic when used in the input mode. We can see that the output signal will turn on when the input power is applied. Like the digital termination panel, each circuit's status is indicated by means of an LED indicator installed on the board. The input circuit is protected by a socketed 3-amp fuse which may be replaced without the need to solder any components. The solid state relay used for this configuration should be a type IAC5 which is available from either Opto-22 or Motorola. Complete details of available relays and their uses on the board are available in the *iCS 930 AC Signal Conditioning/*

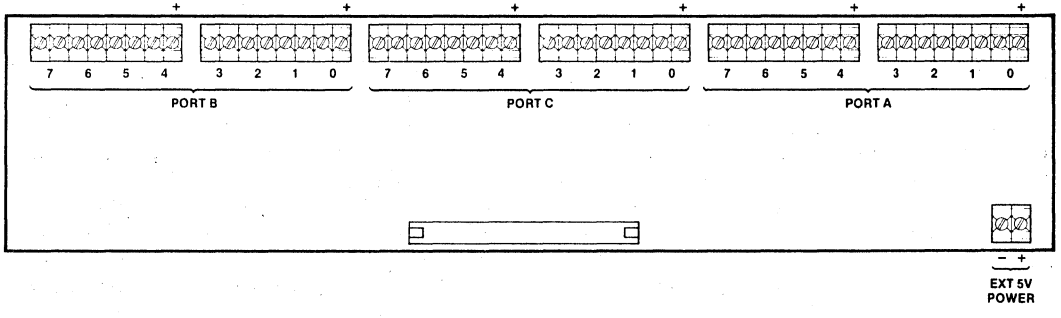


Figure 15. iCS™ 920 Digital Terminator Port Assignment

AFN-01931A

Termination Panel Hardware Reference Manual (manual order number 9800802A). Keep in mind the fact that although this application note represents the solid state relays as being actual relays and contacts, they in fact are solid state and contain no moving parts.

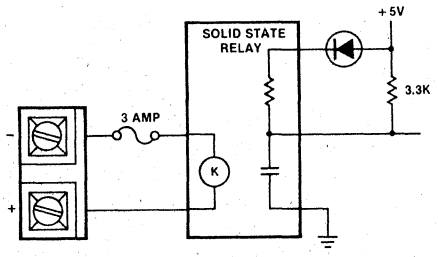


Figure 16. iCS™ 930 AC Terminator Input Circuit

The output configuration is utilized to turn the heater elements (the light bulbs) on and off. Figure 17 provides us with a schematic of the output circuitry. In this case, we will insert a solid state relay of type OAC5 which will handle up to 140 volts RMS at 3 amps. In some cases, it might be desirable to add certain components to the terminator panel when using it in the output mode. Two possible circuit configurations are possible. The first and perhaps the most common will consist of installing a MOV (metal oxide varistor) across the solid state relay contacts. This will be required when the load being driven is inductive in order to prevent the transients generated by the load from damaging the triac in the SSR (solid state relay). Since the SSRs utilize zero voltage switching and the load in our ovens is resistive rather than inductive, our application will not necessitate the installation of this device. The second possibility for additional circuitry also involves driving inductive loads. When the load is highly inductive, a possibility exists that reliable operation of the SSR may not occur because of incorrect values for the dv/dt (a complete description of this phenomenon is available in various publications available from the manufacturers of the solid state relay devices). Provision has been made for installation of an external snubber network should this be required. Again, our oven control system will not require this type of circuitry. Figure 18 is provided for reference should the reader desire to see the location of the additional components on the panel. It should be noted that the component placement does not allow the installation of the MOV and the snubber simultaneously.

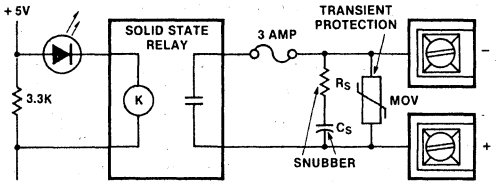


Figure 17. iCS ™930 AC Terminator Output Circuit

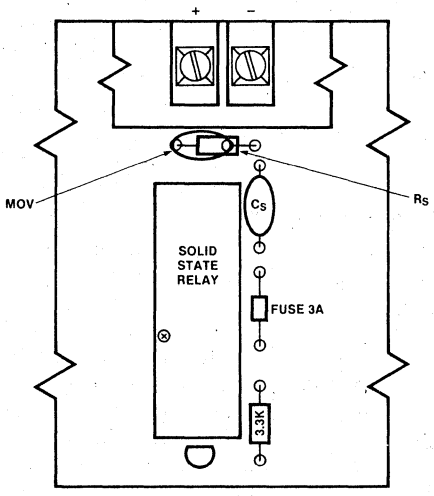


Figure 18. AC Terminator Component Locations

We can now get back to the task of assigning addresses to the various digital channels. The iCS 930 panel has three connector options for connecting it to the computer's I/O ports. The standard configuration utilizes connector J2 to attach the ribbon cable assembly. When this is done, the computer ports A and B will correspond to the 16 channels on the terminator panel (Figure 19). If we look at the termination panel, we will see that there is a provision for the user installation of two additional ribbon connector sockets onto the board. These are used in order to utilize the computer port C. If connector J3 is installed and utilized instead of J2, the channel assignments will be as shown in Figure 20. In a similar manner, connector J1 can be installed and utilized to provide connections between the computer port C and the other eight SSR positions. If we choose the 16 lines required for driving

the indicator lamps from the iCS 920 panel to be ports A and B, then it seems reasonable to assign the eight remaining lines required on the iCS 930 to port C. A feature of utilizing standard ribbon cable assemblies is the ability to easily add ribbon plug connectors to the cable. This will result in an assembly transferring ports A, B and C to the iCS 920 panel (however, port C is not used) and which continues the port C signals to the iCS 930 panel.

Individual channel assignments can now be made, grouping the inputs and outputs together in groups of four (this is done because of a requirement of the single board computers to share terminator and driver component packages in groups of four). Figure 21 provides a drawing showing the channel assignments and the physical wiring locations which will be used to connect the oven heaters and switches.

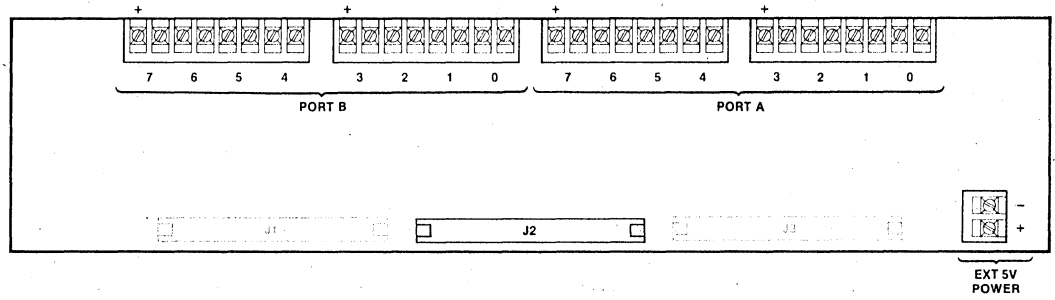


Figure 19. iCS™ 930 AC Terminator Port Assignments

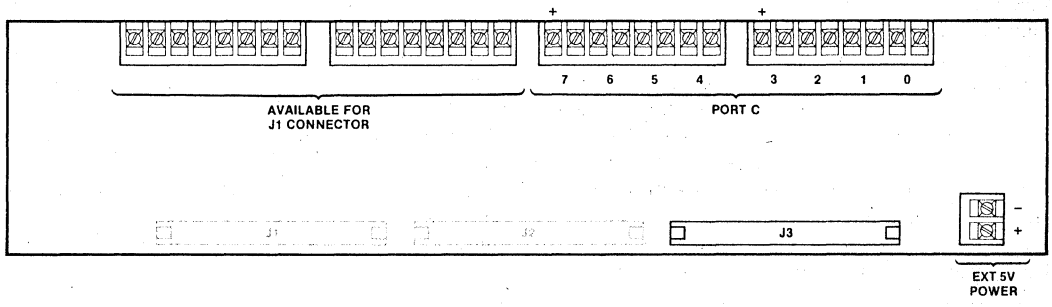


Figure 20. iCS™ 930 AC Terminator Port Assignments

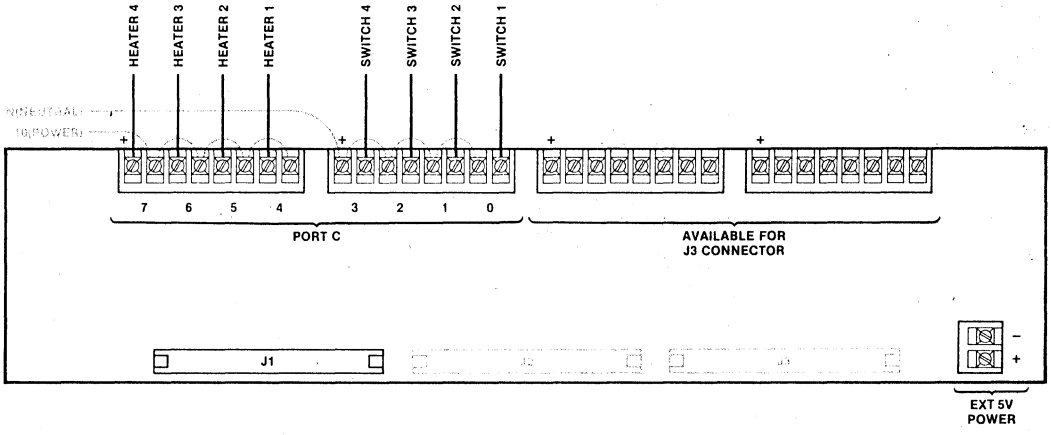


Figure 21. iCS™ 930 AC Terminator Application Configuration

AFN-01931A

Final Channel Assignments

The only task remaining before we have completed our task of assigning channel numbers and physical wire and component locations is to assign these channels on the iCS 920 digital termination panel. Since we have already determined that we will utilize ports A and B, this becomes a simple matter, requiring only an arbitrary assignment of lamp locations using these port bits. The assignments made for one oven can be seen in Figure 22. The entire ladder diagram of the system can now be completed along with port assignments for all signals used. The completed diagram can be found in Appendix B. Note how the port assignments have been shown to the side of the ladder element representing that interface device.

The method used to define a port assignments needs to be clarified since it may not be apparent why a channel of port A was given the address of E80. To begin, we have already indicated that each port consisted of eight channels or bits. We will number these bits from 0 to 7. Since it is possible to have many input/output devices connected to the computer, the possibility exists of having multiple devices which incorporate internally ports A, B, and C. The computer has been designed to support up to 256 of these ports so we have numbered them using the hexidecimal numbering system. The possible port numbers can then range from 00 to FF. It will be found that a common characteristic of most single board computers is the use of assigning the port addresses of E8, E9, and EA to the on-board 8255 parallel peripheral interface. Therefore, the first channel of port A would be defined as having an address of E80; the second channel of port B would be E91, and so forth.

III. SELECTING THE COMPUTER BOARDS

To this point we have delayed the selection of the boards which will be required to provide the computerized control system. The Intel OEM Microcomputer Systems Configuration Guide has been designed to simplify the task of selecting the required system. Our first task is to enter all known information describing our desired system into the project configuration worksheets. These worksheets can then be used to actually select a board configuration which meets our particular requirements. The effort required to accomplish the entry of data is reduced to a minimum through the use of predefined digital and analog configuration worksheets. Our requirement of having a total of 24 parallel data lines, consisting of a mix of high and low level interfaces, can be met by the 24-bit AC/DC combination. Our assignments of requirements for the terminator panels can be made and is shown in Figure 23. It can clearly be seen from the worksheets, that our required interface with the computer digital data will consist of one 24-bit wide connector (had we not used port C assignments, the use of 16-bit wide connectors would have sufficed). This means that our selected single board computer or I/O expansion board must provide at least one edge connector having 24 I/O bits on it.

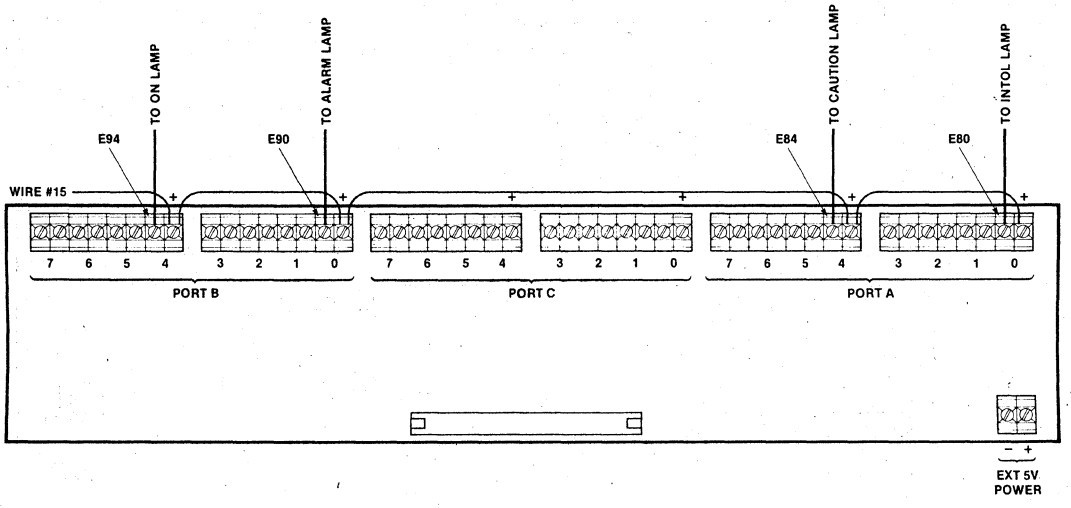


Figure 22. Digitsl Panel Application Configuration

DIGITAL CONFIGURATION WORKSHEET

PROJECT _____

This worksheet will provide the required digital interface configuration data which is required to complete the Project Configuration Worksheet.

Enter Number of Channels

Enter # of Discrete AC Outputs (115-230 VAC)	4	(A)
Enter # of Discrete AC Inputs (115-230 VAC)	4	(B)
Enter # of Discrete DC Outputs (Current > 300 MA)	0	(C)
Enter # of Discrete DC Outputs (Current < 300 MA)	16	(D)
Enter # of Discrete DC Inputs	0	(E)

Compute the Number of iCS 920™ and iCS 930™ Termination Panels

First compute the number of Parallel I/O ports (8-bits each port) required on your iSBC™ board. Round all computations up to the nearest whole integer unless instructed otherwise!

Compute # of iCS 930 Interface Output Ports ((A+C)/8)	1	(F)
Compute # of iCS 930 Interface Input Ports (B/8)	1	(G)
Compute # of iCS 930 Termination Panels ((F+G)/2)	1	(H)
Compute # of iCS 920 Interface Output Ports (D/8)	2	(J)
Compute # of iCS 920 Interface Input Ports (E/8)	0	(K)
Compute # of iCS 920 Termination Panels ((J+K)/3)	1	(L)

Optimization of Digital I/O Port Usage for Minimum I/O Configuration

Compute # of iCS 930 Output "Overflow Channels" DO NOT ROUND OFF) (A+C)/8	QUOTIENT	0	(M)
(Overflow Channels)	REMAINDER	4	(N)
Compute # of ICS 930 Input Overflow Channels (DO NOT ROUND OFF) (B/8)	QUOTIENT	0	(P)
	REMAINDER	4	(R)
Compute # of iCS 920 Output Overflow Channels (DO NOT ROUND OFF) (D/8)	QUOTIENT	2	(S)
	REMAINDER	0	(T)
Compute # of iCS 920 Input Overflow Channels (DO NOT ROUND OFF) (E/8)	QUOTIENT	0	(V)
	REMAINDER	0	(W)
Compute 8-Bit Input Ports Required (P+V)		0	(X)
Compute 8-Bit Output Ports Required (M+S)		2	(Y)
Compute 4-Bit Output Ports Required ((N+T)/4) (ROUND UP)		1	(Z)
Compute 4-Bit Input Ports Required ((R+W)/4) (ROUND UP)		1	(AA)
Compute 8-Bit Port C Requirements ((Z+AA)/2) (ROUND UP)		1	(BB)
Total I/O Parallel Ports Required (X+Y+BB)		3	(CC)
Total # of 24 Channel Parallel I/O iSBC Board Edge Connectors (CC/3) (ROUND UP TO INTEGER)		1	(DD)

Compute Power Requirements for the Termination Boards (DO NOT ROUND OFF)

Compute +5V for iCS 920 Board Outputs (.061 × D)	.976	(EE)
Compute +5V for iCS 920 Board Inputs (.023 × E)	0	(FF)
Compute +5V for iCS 930 Board Outputs ((.020 × (A+C))	.080	(GG)
Compute +5V for iCS 930 Board Inputs (.012 × B)	.048	(HH)
Compute iCS 920 Power Requirements (EE+FF)	.976	(JJ)
Compute iCS 930 Power Requirements (GG+HH)	.123	(KK)

Enter the appropriate data into the Project Configuration Worksheet as shown below:

SEE INSTRUCTION SHEET

PROJECT CONFIGURATION WORKSHEET

EQUIPMENT PARAMETERS:

YOUR REQUIREMENTS / PRODUCT	MEMORY: RAM (Bytes)	MEMORY: EPROM (Bytes)	SERIAL I/O: Serial Ports: TTY	SERIAL I/O: Serial Ports: RS232C	PARALLEL INPUT/OUTPUT: Parallel Lines: I/O	PARALLEL INPUT/OUTPUT: Parallel Lines: OUT	PARALLEL INPUT/OUTPUT: Interrupts: I/O	PARALLEL INPUT/OUTPUT: Interrupts: 24	ANALOG INPUT/OUTPUT: Analog: DI	ANALOG INPUT/OUTPUT: Analog: DI	ANALOG INPUT/OUTPUT: Analog: LOW	ANALOG INPUT/OUTPUT: Analog: HI	ANALOG INPUT/OUTPUT: Analog: HI	ANALOG INPUT/OUTPUT: Analog: OUT	POWER REQUIREMENTS: +5V	POWER REQUIREMENTS: +12V	POWER REQUIREMENTS: -5V	POWER REQUIREMENTS: -12V	COST: Unit	COST: Qty	COST: Qty	COST: Qty
(L) ea iCS-920					(F)	(D)									(JJ)							
(H) ea iCS-930					(B)	(AAC)		(BB)							(KK)							
TOTAL																						
INTEL SOLUTION / SLOT #1																						

Figure 23. Digital Configuration Worksheet

The required power requirements of the termination panels can be calculated using the data provided in the digital configuration worksheet. The information regarding the necessary connectors and the power requirements should then be transferred to the project configuration worksheet (Figure 24).

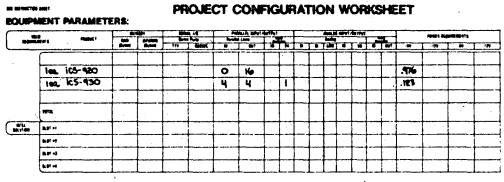


Figure 24.

A similar technique is used to configure the analog signals using the standard analog configuration worksheet as shown in Figure 25. It can be seen that our application will require a single cable connection to a differential input edge connector of an analog input board. The power requirements can be calculated from the current requirements to drive the thermistors and the sensing resistors. The data is entered into the appropriate columns of the configuration tables and then transferred to the project configuration worksheet.

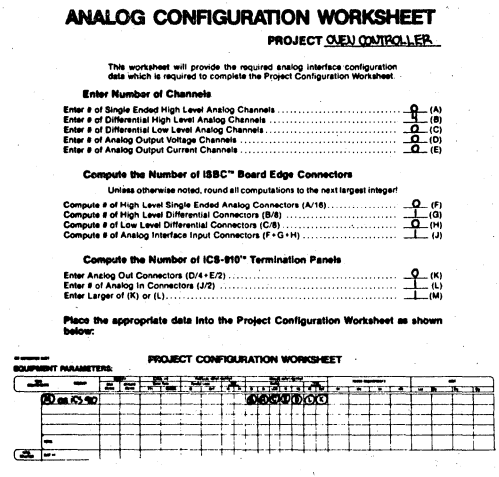


Figure 25.

The only remaining physical element of our control system which we have not defined is the CRT terminal which will be used for setpoint entry and modification. Communications with a terminal requires that we provide a serial RS232C port in our control system. This port requirement is entered onto the worksheet and the system requirements are totaled as shown in Figure 26.

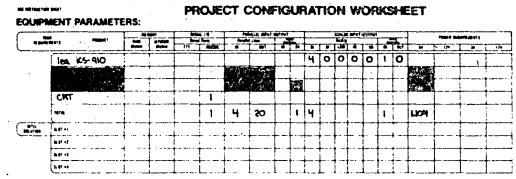


Figure 26.

We must now choose the Intel iSBC boards which will provide a solution to our system requirements. This is done by referencing the summary of key iSBC configuration parameters to find boards which provide the necessary characteristics. Our first task is to choose a single board computer which meets as many of our needs as is practical, while providing performance characteristics adequate to our needs.

Our first requirement for having support for a single RS232C serial communications channel can be seen to be met by a variety of possible boards. Among the possible boards meeting this requirement are:

iSBC 86/12™ iSBC 80/10A™
iSBC 80/20™ iSBC 80/20-4™
iSBC 80/30™

We must look further before a final choice can be made. Again, it can be seen that all candidates also meet the requirement of providing a minimum of one 24-bit wide digital I/O connector. Our decision must be based upon parameters which are not necessarily related to the input or output capabilities. Even though we have not yet developed our software package for our control system, we can safely make some assumptions regarding the completed software package and thus define additional requirements which will enable us to select our desired computer board. The software task will be considerably simplified if we write our programs in a high level language and if we use available drivers for our input and output where they are available. As we will see, the utilization of PL/M and RMX/80™ real-time executive and drivers will make this programming task much less demanding of our time. The trade-off is that these software tools take larger amounts of memory than if we were to write our entire application program in assembly language. Let us make an initial estimate that our system will require about 8K of EPROM and in the neighborhood of 2K of RAM.

Entering this data on the configuration worksheet (Figure 27) enables us to narrow our choice by eliminating the iSBC 80/10A since it does not have sufficient RAM on board.

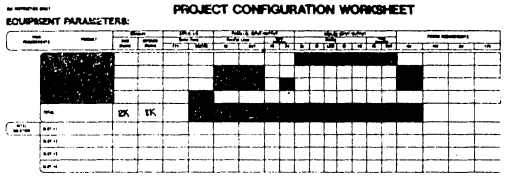


Figure 27.

Since our application is not likely to require extensive math handling capabilities or high speed capabilities, we probably do not need the power found in the iSBC 86/12; so we will remove this product from consideration.

We are now faced with selecting either the iSBC 80/20 board or the 80/30 board for our processor. Each has certain advantages and disadvantages for use in our application. Let's compare these two boards, considering first the iSBC 80/20, then the iSBC 80/30.

iSBC 80/20 board advantages — Slightly lower cost, greater number of I/O lines available.

iSBC 80/30 board advantages — Faster processor, dual ported memory, able to utilize UPI modules.

If the system were to operate in a stand-alone environment and we could be certain that significant expansion would not take place, we would probably choose the iSBC 80/20 computer for our application. If we consider that the system might become a part of a much larger system by future expansions and additions, we should remember that the use of the UPI modules on the iSBC 80/30 computer provides considerable power through multiprocessing capabilities. The dual ported memory can also provide us with the ability to use more sophisticated inter-board communication protocol should the need arise. For the purposes of this application note, we will assume the system is being designed for expansion and we will select the iSBC 80/30 computer.

A good design practice is to provide an extra margin of available memory in the hardware design. Our anticipated RAM memory will use about 2K bytes. The computer will provide us with 4K bytes so we have a considerable margin. This is not true when we look at the amount of EPROM available on the board. Our 8K requirement is identical to the amount of memory available to us on the board. We should consider the use of an expansion EPROM board or the prospect of having to spend a considerable amount of time reworking our program to get it to fit if we find that we have exceeded our estimates. We will select the option of adding a memory expansion board (it can be deleted if we find that our software requirements are less than estimated).

The computer selection and the memory expansion board data can now be entered onto the configuration worksheet as shown in Figure 28. If needed, the addition of the memory expansion board will allow our EPROM requirements to grow up to 16K bytes.

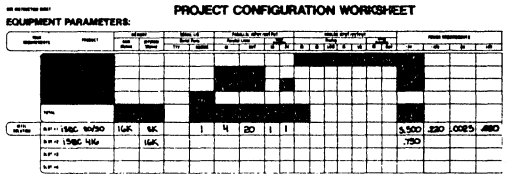


Figure 28.

The only requirement which we have not met is to assign a board to handle the analog input needs of our temperature sensing circuit. The analog voltage can be calculated and will be found to lie in the neighborhood of 4.6 volts at room temperature. This value will increase toward 5 volts as the temperature of the oven increases. Since we have no requirement for any analog output capabilities, we will choose the Intel® iSBC 711™ Analog Input Board to sense the voltage level. This board can be configured to handle a 5-volt full scale input and will provide a resolution of 12 bits. (If an oven requiring a wide range of temperatures and greater resolution were required, we would have to reconfigure our temperature sensor to provide a wider voltage spread over operating temperatures. For purposes of simplicity and clarity we will assume that our temperature resolution is adequate.)

The configuration worksheet can be filled in to reflect the selection of the analog converter and the total power requirements for the system can be computed as has been done in Figure 29. We now need to select a chassis and power supply in order to complete the application hardware design phase.

The Industrial Chassis

Before the boards can be operated together to form a control system, a means of allowing communica-

SEE INSTRUCTION SHEET

PROJECT CONFIGURATION WORKSHEET

EQUIPMENT PARAMETERS:

YOUR REQUIREMENTS	PRODUCT	MEMORY RAM (Bytes)	MEMORY (E)PROM (Bytes)	SERIAL I/O Serial Ports TTY	SERIAL I/O Serial Ports RS232C	PARALLEL INPUT/OUTPUT Parallel Lines IN	PARALLEL INPUT/OUTPUT Parallel Lines OUT	Input Connectors 16	Input Connectors 24	ANALOG INPUT/OUTPUT Analog DI	Analog SI	Analog LOW	Analog IO	Analog VO	Analog Connectors IN	Analog Connectors OUT	POWER REQUIREMENTS +5V	POWER REQUIREMENTS -12V	POWER REQUIREMENTS .5V	POWER REQUIREMENTS .12V	COST List	COST Qty	COST Qty	COST Qty
	iSBC 86-30									4	0	0	0	0	1	0								
	iCS-80					0	16										.896							
	iCS-930					4	4		1								.125							
	CRT				1																			
	TOTAL	2K	8K		1	4	20		1	4					1		1.021							
INTEL SOLUTION	SLOT #1 iSBC 80/30	16K	8K		1	4	20	1	1								3.500	.880	.0025	.050				
	SLOT #2 iSBC 416		16K														.750							
	SLOT #3 iSBC 711									8					1		1.700							
	SLOT #4																							
	SLOT #5																							
	SLOT #6																							
	SLOT #7																							
	SLOT #8																							
	SLOT #9																							
	SLOT #10																							
	SLOT #11																							
	SLOT #12																							
	ROM/EPROM																.090							
	I/O DRIVERS TERMINATION																.420							
	USER SUPP HARDWARE																							
	SUBTOTAL																8.668	.220	.0025	.050				

NOTE:
DI-Differential Input
SI-Single ended Input
IO-Current Output
VO-Voltage Output

SYSTEM INTEGRATION	SLOTS AVAILABLE	POWER AVAILABLE +5V	POWER AVAILABLE -12V	POWER AVAILABLE .5V	POWER AVAILABLE .12V
SYSTEM					
CHASSIS iCS 80-35	4	14.0	2.0	.900	.300
CARD CAGE					
POWER SUPPLY					
				TOTAL SYSTEM COST	

Figure 29.

tion between the boards and of distributing power among the boards must be found. This requirement is met by specifying a chassis into which the boards will be mounted. The Intel® iCS 80™ Industrial Chassis provides an environment for operating the boards which is specifically designed to operate in an industrial area.

The chassis has been designed to facilitate mounting into either a standard 19-inch RETMA cabinet or it may be rear-panel mounted into an enclosure such as may be found in applications requiring the use of a NEMA electrical enclosure. The card chassis has been mounted in such a manner as to hold the single board computers and expansion modules vertically, facilitating maximum cooling of the boards. Fans are provided to aid the normal convection cooling process. Card racks may be installed into the iCS 80 chassis to expand the card support capability to a maximum of 12 card slots in groups of four. Either an iSBC 635 or 640 power supply can be mounted into the industrial chassis to provide power up to 4 or 12 boards capability, respectively.

Our application design requires the installation of a three board solution, so we will choose the iCS 80 chassis with one iSBC 635™ power supply. We will choose to mount our control system in a standard NEMA 12 enclosure to protect the unit from the industrial environment. We should refer to the *iCS 80 Industrial System Site Planning and Installation Guide* (manual order number 9800798) for complete details for selecting appropriate enclosures and installation instructions.

The +5 volt power needed to support the various termination panels and to supply a reference voltage for the thermistors is available from a barrier strip located on the lower front of the iCS 80 chassis (Figure 30). Our wiring can be routed to this barrier strip for those circuits requiring either 5-volt DC or the system logic common. A fuse holder is provided and a fuse should be installed for system protection. We will install a 2-amp fuse into the holder (our maximum power requirement for external circuitry should be 1.22 amps according to Figure 26).

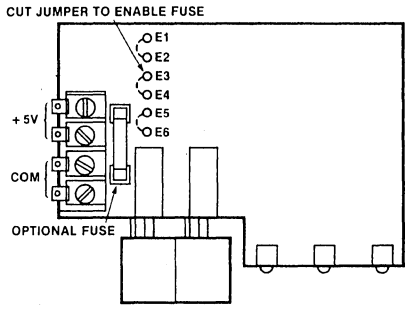


Figure 30. Industrial Chassis DC Power Strip

The remaining terms required in our ladder diagram (Appendix B) consist of a high voltage neutral and a source of switched high voltage power for the heater lamps. Both of these terms are available from the iCS 80 industrial chassis. It is desirable to utilize the same switched power for both the computer system and our external signals, so that we can provide protection to operators when one portion of the system is shut down. A common source will insure that all portions of the system are inactivated if repair is being done. The iCS 80 chassis incorporates a heavy duty industrial key-lock switch for its power switching. The outputs of this switch are available to the user at a terminal barrier strip located on a fold-out panel on the rear of the chassis assembly (refer to Figure 31). We can see that our neutral wire should be connected to terminal 5 (filtered AC low) and the wire for the AC high, wire #10 on the ladder diagram, should be connected to terminal 9. This will provide us with a switched, fused, and filtered power source for our external wiring.

As we will be installing the chassis into a NEMA enclosure, we will not want to use a standard power cord since this would involve the additional expense of installing a duplex outlet in the cabinet. The power wiring can be installed directly onto the power barrier strip by placing the AC hot wire on barrier number 1, the neutral wire onto barrier number 4, and the ground onto barrier number 3.

The hardware implementation of the system can now be considered to be complete. Before the system can function as a control for the oven temperatures, we must define the relationships between the various pieces of the oven system and we must also define the operator interface with the CRT terminal. Thus, we begin the software phase of our design.

IV. DETERMINATION OF SOFTWARE APPROACH

The task of providing the relationships between the various system components falls into the category of writing the software. Before we actually begin to develop this software, we will define certain guidelines which can be used to organize and simplify the task.

Let us consider the general environment under which our programs will operate. We find that we have essentially two choices in this area. First, we can consider the entire process as a sequential set of predefined operations in which we must perform each operation before moving to the next until finally we complete the sequence and begin again. (This is analogous to using a single stepper switch to design our control system.) Since each oven is independent of the others, we can not afford to use

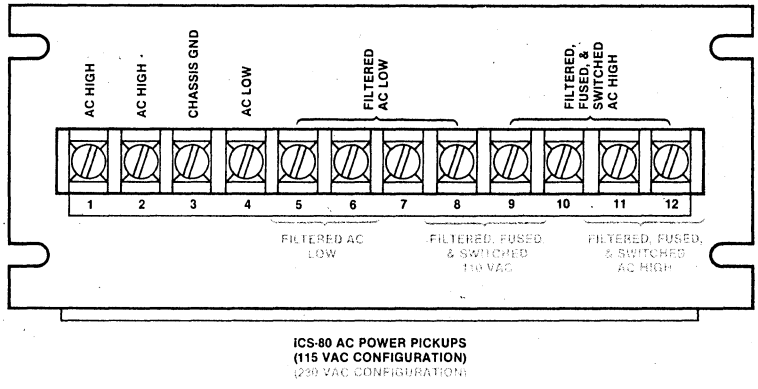


Figure 31. Industrial Chassis AC Power Strip

this approach since we could get tied up waiting for something to happen in a particular oven and would have to ignore the other ovens. The designer familiar with relay design will probably be thinking, at this point, that we should use a separate sequential operation for each oven or device to be controlled. Indeed, this is exactly what we can do with our software by using what is known as a real-time executive. This tool will allocate the computer's resources in such a manner as to provide us with the capability of having independent software programs or tasks operating at what appears to be the same time. We will make our first assumption that our software will be written using such a tool and we will specify that we will operate under Intel's RMX/80 Real-Time Multi-Tasking Executive. We will discuss more detail of this software tool as we develop our programs.

Next, we must consider the language which we will use to actually define our required operation. We have many alternatives from which to choose. Let us look at several of the alternatives in some detail.

Assembler

Assembler language is probably the most basic tool with which we can program a computer. It is considered to be the most efficient user of program memory and processor time. These features are made possible because each assembler instruction line is converted directly into a corresponding machine instruction. From a programming standpoint, assembler language is the most difficult to use since any task must be defined by subdividing that task into a multitude of smaller operations compatible with the available instructions of the computer. To use this language, we must be familiar with the architecture of each computer with which we desire to operate. The use of the language is somewhat simplified through the use of an Intel supplied assembler which converts the assembler code into machine instructions and provides listings of the operations which have been entered. A complete description of the Intel 8080/8085 Assembler Language is available in the *8080/8085 Assembly Language Programming Manual* (manual order number 9800301B).

The user should consider this programming tool when his application requires the minimum amount of memory (such as might be required for very large volume designs where memory cost is a factor) or where a highly time dependent routine must be defined. Our oven application does not fall into either of these categories, so we will choose not to use this language in our instance.

PL/M

Intel's PL/M language offers an efficient, structured, high level systems programming language. Before proceeding, let us be clear on the benefits of using a high level language. First, the use of high level languages results in reduced development time and cost. High level languages provide the ability to program in a natural algorithmic language. In addition, they eliminate the need to manage register usage or to allocate memory. Second, high level languages provide improved product reliability because programs tend to be written in structured formats and result in a minimum of extraneous branches which might cause testing problems. Finally, their use produces programs which are better documented and are easier to maintain.

On the other hand, high level languages do not optimize the code segments as well as can be done by an experienced assembly language programmer. As a result, most compilers (routines which convert the high level languages into machine executable code) use more program storage than those written by the assembly language programmer. Different languages and compilers require different amounts of memory for the same task.

PL/M-80 is probably one of the most efficient high level languages for use on microcomputers. It has been determined that PL/M-80 users can expect to use between 1.1 to slightly more than 2 times as much program memory as would be used for the same task written in assembly language. For this reason, we must place the use of this language high upon our list of possible languages in this application.

A glance at the *PL/M-80 Programming Manual* (manual order number 98-268B) indicates that the language is highly structured and seems to lend itself very well to handle logical type operations. It seems to have the greatest weakness in its math handling capabilities in that it does not support negative numbers or fractions. It is reasonable to assume that the oven application can be handled entirely with positive integer numbers so this limitation will not unduly hamper our use of this language. We will keep these features in mind when making a final decision.

FORTRAN

Intel's FORTRAN-80 provides the full subset of ANSI FORTRAN 77. In many cases FORTRAN-80 has features that exceed the specifications for both the subset and the full versions of FORTRAN 77. Most of the power of this language lies in its ability to easily handle complex mathematical expressions. Obviously, it does not have any limitations regarding fractions or sign of the numbers involved. It should be used when the application requires the use of mathematical computations. The power of the language, however, means that the use of the language will take a heavy toll of memory allocation. A complete description of the FORTRAN version supported by Intel and its use on the iSBC computers can be found in the *FORTRAN-80 Programming Manual* (order number 9800481A) and in the *ISIS-II FORTRAN-80 Compiler Operator's Manual* (order number 9800480).

It is unlikely that the magnitude of mathematical routines required to control the temperature of our ovens will be complex enough to justify the use of FORTRAN. Keep in mind that, if such a situation were encountered, it is feasible to use a combination of programming languages to create our final module.

BASIC

Certaintly the most well known high level programming language today is BASIC. It offers a quick way of applying the computational capabilities of the computer to a wide range of applications. The Intel RMX/80 BASIC-80 is an interpreter designed to operate with Intel's single board computers and contains extended disk handling capabilities. As an interpreter, it differs from other high level languages in that it results in a relatively slower operating solution to an application. It is also not possible to use BASIC to generate multiple independent tasks which can compete for computer resources.

For these reasons, we cannot consider the use of BASIC for a solution to our application.

Final Selection of Language

From the above discussion, it seems clear that our choice for the application being demonstrated is to use PL/M-80 as our programming language.

With this in mind, we can begin the task of actually generating the code which will complte our application and provide an operating control system.

V. DEFINING SOFTWARE TASKS

The software implementation can begin as soon as we have broken our control functions into independent "tasks". We can then handle each task separately as though it were the only thing which had to be done by the control system. In the event that we find that one of our tasks must communicate with or be interlocked with another, we will handle this need through the use of "exchanges". The "exchange" can be thought of as a mailbox into which messages are deposited and picked up by the various tasks. These messages convey the necessary information between the otherwise independent programs. When all tasks have been coded, we will combine them using the facilities of RMX/80.

Our oven application can be broken down into three functional areas or tasks. These are:

1. The Control Task which will be used to actually sense the oven temperature and to provide the required responses to the heaters and the indicator lamps.
2. The CRT Update Task will be used to provide a "snapshot" of the system operations to a person viewing the CRT terminal.
3. The Parameter Update Task will be used to examine and update the oven setpoints and tolerances.

The choice of these three tasks has been essentially arbitrary in nature. Certainly, other choices and groupings of functions could easily have been made. We will use these choices for our example and will proceed with our development accordingly.

We have two other supporting tasks which must be included in our system. Fortunately, these tasks are predefined and fully supported within RMX/80's libraries; thus we need not write these functions. The two supporting tasks are:

4. A Terminal Handler Task to support the actual interface to the CRT terminal. It provides echo of input characters and signals when data is ready to be read. It will output messages to the terminal and signal when all characters requested have been sent.
5. An Analog I/O Driver Task to request and handle the handshaking which is required to communicate with the analog input board. It will signal us when data has been input and is available for use by our user written tasks.

We can proceed with the implementation of each of our three tasks which we have defined. The first step with each will be to develop a flowchart which shows the required operations to implement that task. This flowchart will show any intertask communications or exchanges that may be required tunately, the voltage and temperature are not using the facilities provided by our programming language.

Oven Control Task

The sequence of operations required to perform the control task can be defined using the flowchart shown in Figure 32. Let us examine the required steps in more detail.

An arbitrary decision has been made to only sample and control the ovens once each second. This will allow some time for the system to respond once a heater output has been set. The first step in our control task is to wait for one second to elapse.

Our next subtask should be to read the status of the various oven control switches on the operator's control panel. This item could wait until a later time, but there is no harm in handling it at this time.

Next, we see a block indicating the input of data regarding the current oven temperatures. This oven temperature data will certainly be used by the task handling the snapshot display on the CRT so we must give some consideration to the validity of the data. While we are in the process of getting the data and converting it to engineering units (next step), there will be periods during which the stored temperature data does not reflect the actual oven temperature. An example might be when we are actually moving the 16 bits of the temperature since we can only move data 8 bits at a time. During this period, we would not want another task to use the data and since each task is going to operate independent of others, we must provide some type of lockout of the data while we are operating on the temperatures (an alternative would be to have each task get its own temperature from the A/D converter and convert it to engineering units, but this would seem to waste memory and computer time). We can provide this lockout by creating an exchange to communicate with other tasks. If we make a message available in this exchange when the data is valid and cause no messages to be available when the data is nonvalid, we can effectively lock out tasks from using the data when it is in the process of being updated. This is done by requiring those tasks to test for the presence of a message at the exchange before they get the temperature data. If no message is present, they must wait until one is placed into the exchange before proceeding. Just before we update the temperatures we will fetch the message from the exchange, leaving it empty while we work on the data. later we will again restore the message when the update is complete.

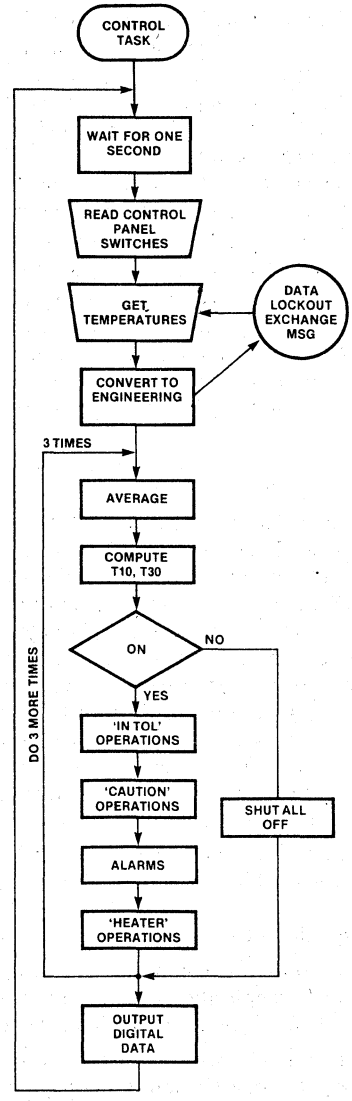


Figure 32. Control Task Flowchart

The number obtained from the analog converter provides us with a value which is proportional to the temperature of the oven. Our next step is to convert this number into engineering units. Unfortunately, the voltage and temperature are not related in a linear fashion since the thermistor is a nonlinear device. We will have to develop a technique to obtain a corrected value. For the purposes of this application note and in an attempt to keep the application as simple as possible, we have chosen to utilize a single table look-up to perform this conversion. Alternatives might have been to utilize FORTRAN routines to mathematically perform the conversion or to have separate tables for each oven. Once the conversion has been made, we must return a message to the data lockout exchange to allow other tasks access to the data.

Because we must deal with four ovens, the operations related to each individual oven must be performed four times, once for each. This is easily handled as we will see, since PL/M is a block structured language. Our flowchart need only remind us that the operations need be done four times.

The next step has been defined as performing some digital filtering of the temperature by averaging the current temperature with the temperature of one second ago. This filtered value will be used to perform subsequent computations and to make future decisions.

We have defined earlier in our definition of the control algorithm that we would use a derivative control. We have chosen to project the temperature ahead for a period of 10 and 30 seconds. We must calculate the rate of change and the temperatures in 10 and 30 seconds so that this data will be available when needed.

Now that the calculations have been made to determine numeric values required for the decision making process, we must begin the process of determining the status of each indicator and oven heater. A test will be made of the oven run switch and if it is found to be turned off, we will turn off all indicators and the oven heater associated with that oven. If the switch is found to be turned on, we will set the status of the "in tolerance", "caution", and "alarm" indicators according to our oven control algorithm. The oven heater will be turned on or off according to the projected temperature in 30 seconds.

Rather than output the individual oven indicator and heater data four times (once for each oven), we will perform the computations associated with making the decision four times (this saves code since we can use the same program steps with only pointers being exchanged). At the end of this time, a single operation will output the data to all ovens and indicators at the same time. Outputting to a computer port will actually cause the device to turn on or off according to whether the output bit is a one or zero.

We will then return to the beginning of our task to wait until another second elapses before we again perform the indicated functions.

Control Task Source Coding — The coding of our tasks is a straightforward procedure once we have prepared a flowchart. Since we are using PL/M-80 and RMX/80, the coding sequence for a task will be as follows:

1. Define any variables or structures which will be used in the module. This involves providing information defining variables as being either an 8 or 16-bit variable and declaring if that variable is to be a part of the task being coded or is to be found in some other task. If any arrays or structures are to be used, they must also be defined. Finally, if any program locations are to be used, they must be declared.
2. The task must be initialized. That is to say that any assumptions which will be made as to initial data values in subsequent instructions must be initially forced to this initial value.
3. The actual task must be coded to match the operations called out in the flowchart.

We will look at some examples of this coding process using the control task flowchart. The complete listing of this module and all modules actually used to provide the oven control system can be found in Appendix C.

At first glance, it would seem that the listing is extremely complex, but as we will see it is made up of straightforward pieces. The listing is made up of three parts as we have mentioned above when defining the steps required to generate a program. The first part (line numbers 1 through 50) is used to define parameters, variables, and external elements. The general types of elements making up this portion fall into typical categories. The first general category consists of DECLARE statements. Examples of typical lines will help explain their meanings (when actually developing the program, this first section was created piecemeal by

making an entry when it was found that a need for that term existed as the execution code in sections two and three were written).

Examples of the "declare" statement are shown below. For example, on line 11 we find:

```
11  1     Declare (n,k) byte;
```

This means that the variables "n" and "k" are being defined as terms which represent numbers or data which is one byte or 8 bits wide. The "11" is the program line number, and the "1" indicates that we are in the first level of nesting.

We can also see the use of the "literal" expressions such as used in line 4. The expression:

```
4  1     DECLARE FALSE LITERALLY '00H';
```

means that we are creating a new instruction called "false" and that its meaning is to be interpreted by the compiler as being equivalent to the value of zero.

Rather than dwell on the declaration, let us move on to the coding process which was used to generate the actual program. Keep in mind that the use of PL/M-80 requires that all terms used be declared in the program module. Refer to the *PL/M-80 Programming Manual* (order number 9800268B) for a full description of the PL/M language.

Program Initialization — The initialization portion of the program can be found on lines 51 through 59 of the control task program listing. This section is used to initialize data and to provide known entry conditions before we enter the repetitive program loop. This code is only executed when the system is reset or when the power is turned on. The control task requires two types of initializations; one to initialize the computer's output port and the other to set up the A/D converter. The requirements for each can be found in the RMX/80 User's Guide and the *iSBC 80/30 Single Board Computer Hardware Reference Manual* (order number 9800611A). Actual instruction examples are given in these manuals for the initialization operations.

Program Body — The program which actually provides the control operations can be found on lines 60 through 126 of the program listing for the control task. It has been divided into sections which correspond directly to the flowchart that was prepared earlier. Most instructions in PL/M-80 language follow closely the English structure which describes what is being done. The exceptions generally follow definite predefined formats. The format such as used on line 61 to wait for one second to elapse is an example of one such exception. Any time we desire to wait for a definite time period, we use an instruction of the form:

```
MSG$PTR = RQWAIT (.DUMMY$EXCH, TIME DELAY);
```

Whatever time delay we wish to use is expressed in increments of 50 msec time periods. Our example requires a time delay of one second so we will use the delay notation of 1.0/0.050 = 20 time units (this command is actually calling upon the RMX/80 executive to handle the delay).

The oven enable switch data has been defined by us to be routed by the hardware to the computer port "EA" which converts to a decimal number, 234. If we define an internal memory location for this data and call it BLOCK0, then we can get the oven switch data by using an input statement. Since the data sense is inverted through the hardware, we can provide meaningful internal data if the signal is re-inverted as it is loaded into memory. The instruction on line 62 of the control task listing performs this task.

We are now ready to get the analog data from the A/D converter. Our flowchart shows that we must lock out the other tasks from access to the temperature data during this time period, so we must first remove the enable message from the exchange in which it is stored. Messages are removed from an exchange by using an instruction of the form:

```
STORAGE = RQWAIT (EXCHANGE NAME,0)
```

Line 63 of the program listing means that we will get a message from our storage exchange which is called "Temp$lockout$exch" and store it in a memory storage area called "Lockout". Now, no other task can get a message from this exchange since it is empty, so it is permissible to operate on the temperature data. (Note how similar this command is to the one used to wait for a delay. Indeed, this is the same request for RMX/80, but it requests a time delay of zero.)

During the initialization, we built a message defining the characteristics of the analog signals and of the analog conversion board which we are using. Remember that we have indicated that the task of getting this data from the board is provided to us by one of RMX/80's predefined drivers. All that is necessary at this time is to inform that driver of our desire to get data, then wait until it has done its job and the data is available for us. The actual communication between our applications task and the analog driver is done using the idea of an exchange similar to that we have used to lockout the data.

A flowchart can now be prepared showing the steps required to implement the CRT update task. This flowchart is shown in Figure 34. The coding of the program to support this task can be found in Appendix C. The development is identical with that which we described in the sections regarding the control task. Again, the software is divided into three parts, the declaration statements from line 1 to 81, the initialization on lines 82 to 87, and the actual task code on line 88 to 207.

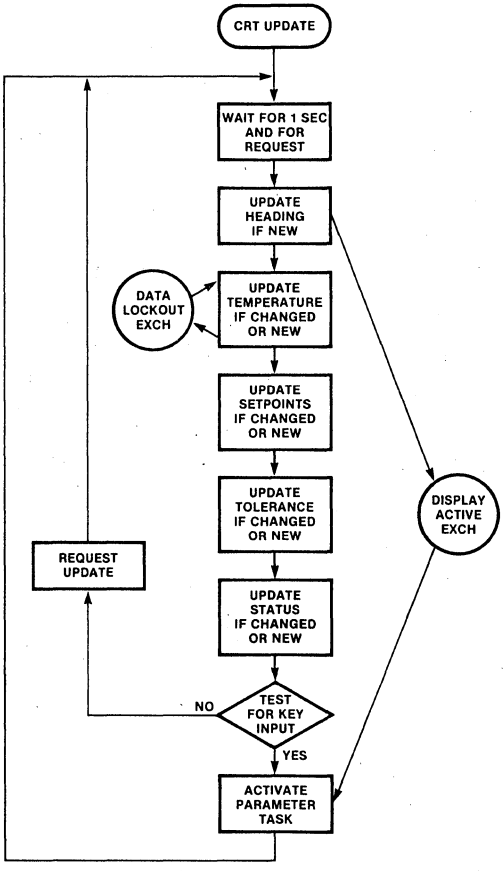


Figure 34. CRT Status Flowchart

A technique to exit from the CRT update mode and to get into a mode which will allow modification of the parameters has been introduced into the program and the display format. This is in the form of a message on the botton of the screen requesting the entry of an escape character to adjust setpoints. The software has been written in such a manner as to test for a character inut from the keyboard and if one is found corresponding to that character, the update task will allow the parameter update task to take control of the terminal (lines 190 to 204 of the listing).

Parameter Update Task

The parameter update task is used to actually allow the modification of the setpoints and the tolerances associated with each oven. A second use of the task is to provide a tool for establishing the zero offset associated with each analog channel so that an offset into the temperature linearization table can be computed by the control task.

Figure 35 shows the flowchart which describes the steps required to perform these operations. When the task has been completed, we will return to the CRT update task.

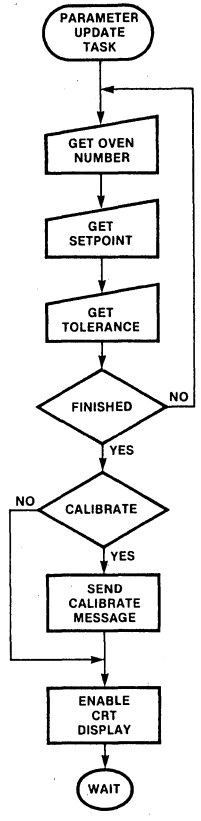


Figure 35. Parameter Update Flowchart

The program code for this task can be found in Appendix C and again follows the formats which we have discussed earlier. No attempt will be made in this document to provide a narrative of the listing since it follows the flowchart in development.

Support Programs

Three subprograms (procedures) have been written which provide functions which are common to the three tasks. This has been done to minimize repeating code segments thus saving as much memory as possible. These three subprograms support:

1. Conversion of a decimal string from the terminal into a binary number. This program is called ASC2BINARY and can be found in Appendix C.
2. Storage for common variables used by more than one task. These variables could easily have been included in other tasks but a purely arbitrary decision was made to include them in a separate module.
3. Conversion of binary numbers into a decimal string suitable for output to the terminal. This program is called DEC$REP and is found in Appendix C.

We now have completed the coding of the software to support our oven application. We must finish by combining all the software together to form a single loadable module.

VI. FINAL IMPLEMENTATION

When all code was linked and loaded to form an executable program module, it was found that the system required 9,041 bytes of EPROM and 1,735 bytes of RAM. These values fall within our hardware capabilities and will rquire that we program and insert nine EPROMs into the EPROM expansion card.

The system can now be tested and installed to control the ovens of our application. The actual system described in this application note has been constructed and tested. It has been found to control the oven temperatures of four ovens and performs as we anticipated when we developed our control strategy earlier in this application note.

VII. CONCLUSION

We have shown how Intel's single board computers, industrial chassis, termination panels, and software can be configured to provide a solution to a typical control application. We have seen how the development of a solution to a control problem can proceed along a predetermined and logical path. Truly, the utilization of the microprocessors can lead to optimum and cost effective solutions to control applications.

We will send a message to the analog driver telling it what we want it to do, then we will wait until it sends a message back to one of our exchanges telling us that it is done. The format for sending a message to an exchange always follows the form:

CALL RQSEND (EXCHANGE NAME, MESSAGE NAME);

Line 64 of the listing shows that we have requested the input of the analog data since we have sent our message, Convert, to the analog driver's exchange which is called RQAIEX. We will wait until the operation is complete by using the line of code shown on the listing line 65. This is the same operation type that we used to get our message back providing a lockout earlier. The program will wait until a message is available before continuing.

The data must now be converted into engineering units. We earlier indicated that we would use a table lookup to perform the linearization, so we have included this table as a part of our program at line 50. The offset into the table corresponding to our temperature must be determined so that the correct value can be stored. Because we have four ovens, we will perform the operation four times with the data each time corresponding to the appropriate oven. These operations can be followed on lines 77 through 81 of the listing.

Lines 67 through 76 are used to establish an offset to be applied to the analog temperature data when the system is running. This program is only designed to be used during the start-up operations and is activated when a message containing a calibration request and current temperature is sent to its exchange.

The temperature lockout must be removed to enable other tasks to use this data. This is done on line 82 by sending the message back to the exchange used for intertask lockout communications.

The remainder of the program follows the flowchart and the operations can be followed using a flowchart and the listing. Each element of the flowchart corresponds to a block of code on the listing.

CRT Update Task Development

Earlier, we stated that the CRT update task would be used to allow the operator to view a "snapshot" of the four ovens. Let us turn our attention to developing the software which is required to accomplish this. We can begin by defining the elements which we feel should be displayed, then defining the format to actually be used with the CRT terminal.

Obviously, we need to provide the current temperature of each oven on our display screen. If we display the actual temperature, it seems reasonable to assume that we should also show the setpoint so that a determination can be made as to how well the system is performing. The control algorithm has been defined to use an allowable range to determine system outputs, so it would seen wise to also show this parameter. Finally, we should inform the viewer of the status of the oven so that he will realize that the reason an oven temperature is low is because the oven is off rather than an oven malfunction. Other items could be added if desired by the system designer, depending upon the total system requirements or the characteristics of the users.

We can now prepare a drawing of the CRT display to generate a layout of our desired characters and to generate an aesthetic display for viewing during operation. This drawing can be found in Figure 33.

Several techniques are available to output the required displays to the terminal. A decision must be made as to the frequency of screen updates; will we constantly refresh the data or do it only at certain intervals of time? If the terminal has the ability to disable the cursor, it makes sense to update data continuously. If the cursor cannot be disabled, its movement tends to be distracting, so the updates should be kept to a minimum. The terminal used for the application note did not have a disable feature, so we will make the decision to only update the screen once each second.

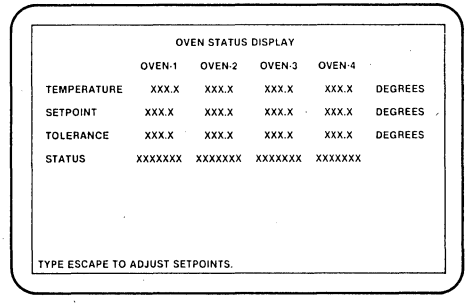


Figure 33. CRT Status Display Layout

The decision to delay updates leads us to make another decision regarding the screen updates. If we only update a line which has data which has changed since the last update, the cursor movements will be kept at a minimum since it is unlikely that all parameters will ever change each second.

APPENDIX A
SELECTED DATA SHEETS

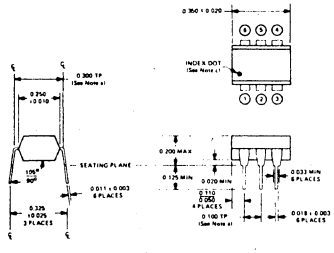
TYPES TIL113, TIL119 OPTO-COUPLERS

BULLETIN NO. DL-S 7312032, NOVEMBER 1973

- Gallium Arsenide Diode Infrared Source Optically Coupled to a Silicon N-P-N Darlington-Connected Phototransistor
- High Direct-Current Transfer Ratic . . . 300% Minimum at 10 mA
- Base Lead Provided for Conventional Transistor Biasing
- High-Voltage Electrical Isolation . . . 1500-Volt Rating
- Plastic Dual-In-Line Package
- Typical Applications Include Remote Terminal Isolation, SCR and Triac Triggers, Mechanical Relays, and Pulse Transformers

mechanical data

The package consists of a gallium arsenide infrared-emitting diode and an n-p-n silicon darlington-connected phototransistor mounted on a 6-lead frame encapsulated within an electrically nonconductive plastic compound. The case will withstand soldering temperature with no deformation and device performance characteristics remain stable when operated in high humidity conditions. Unit weight is approximately 0.52 grams.



NOTES:
a. Leads are within 0.005 radius of true position (TP) at the gauge plane with maximum material condition and unit installed.
b. All dimensions are in inches unless otherwise noted.
c. Pin 1 identified by index dot.
d. Terminal connections:
   1. Anode } Infrared-emitting diode
   2. Cathode }
   3. No internal connection
   4. Emitter
   5. Collector } Phototransistor
   6. Base (For TIL119, make no external connection) }

absolute maximum ratings at 25°C free-air temperature (unless otherwise noted)

Input-to-Output Voltage	±1.5 kV
Collector-Base Voltage (TIL113)	30 V
Collector-Emitter Voltage (See Note 1)	30 V
Emitter-Collector Voltage	7 V
Emitter-Base Voltage (TIL113)	7 V
Input-Diode Reverse Voltage	3 V
Input-Diode Continuous Forward Current at (or below) 25°C Free-Air Temperature (See Note 2)	100 mA
Continuous Power Dissipation at (or below) 25°C Free-Air Temperature:	
Infrared-Emitting Diode (See Note 3)	150 mW
Phototransistor (See Note 4)	150 mW
Total (Infrared-Emitting Diode plus Phototransistor, See Note 5)	250 mW
Storage Temperature Range	−55°C to 150°C
Lead Temperature 1/16 Inch from Case for 10 Seconds	260°C

NOTES: 1. This value applies when the base-emitter diode is open-circuited.
2. Derate linearly to 100°C free-air temperature at the rate of 1.33 mA/°C.
3. Derate linearly to 100°C free-air temperature at the rate of 2 mW/°C.
4. Derate linearly to 100°C free-air temperature at the rate of 2 mW/°C.
5. Derate linearly to 100°C free-air temperature at the rate of 3.33 mW/°C.

electrical characteristics at 25°C free-air temperature

PARAMETER		TEST CONDITIONS†			TIL113			TIL119			UNIT
					MIN	TYP	MAX	MIN	TYP	MAX	
$V_{(BR)CBO}$	Collector-Base Breakdown Voltage	I_C = 10 μA,	I_E = 0,	I_F = 0	30						V
$V_{(BR)CEO}$	Collector-Emitter Breakdown Voltage	I_C = 1 mA,	I_B = 0,	I_F = 0	30			30			V
$V_{(BR)EBO}$	Emitter-Base Breakdown Voltage	I_E = 10 μA,	I_C = 0,	I_F = 0	7						V
$V_{(BR)ECO}$	Emitter-Collector Breakdown Voltage	I_E = 10 μA,	I_F = 0					7			V
$I_{C(on)}$	On-State Collector Current	V_{CE} = 1 V,	I_B = 0,	I_F = 10 mA	30	100					mA
		V_{CE} = 2 V,	I_F = 10 mA					30	160		
$I_{C(off)}$	Off-State Collector Current	V_{CE} = 10 V,	I_B = 0,	I_F = 0			100			100	nA
h_{FE}	Transistor Static Forward Current Transfer Ratio	V_{CE} = 1 V,	I_C = 10 mA,	I_F = 0		15,000					
V_F	Input Diode Static Forward Voltage	I_F = 10 mA					1.5			1.5	V
$V_{CE(sat)}$	Collector-Emitter Saturation Voltage	I_C = 125 mA,	I_B = 0,	I_F = 50 mA			1				V
		I_C = 10 mA,	I_F = 10 mA							1	
r_{IO}	Input-to-Output Internal Resistance	V_{in-out} = ±1.5 kV, See Note 6			10^{11}			10^{11}			Ω
C_{io}	Input-to-Output Capacitance	V_{in-out} = 0,	f = 1 MHz,	See Note 6		1	1.3		1	1.3	pF

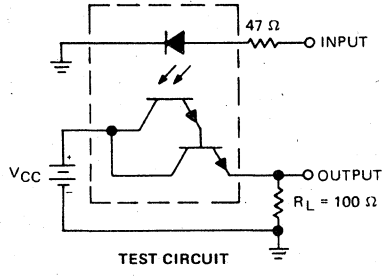
NOTE 6: These parameters are measured between both input-diode leads shorted together and all the phototransistor leads shorted together.

† References to the base are not applicable to the TIL119.

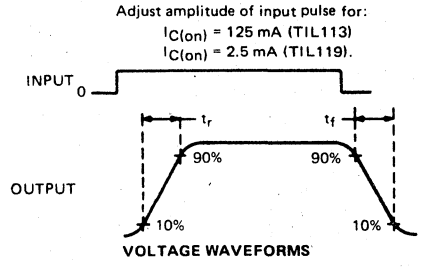
switching characteristics at 25°C free-air temperature

PARAMETER		TEST CONDITIONS		TIL113			TIL119			UNIT
				MIN	TYP	MAX	MIN	TYP	MAX	
t_r	Rise Time	V_{CC} = 15 V,	$I_{C(on)}$ = 125 mA,		50					μs
t_f	Fall Time	R_L = 100 Ω,	See Figure 1		50					
t_r	Rise Time	V_{CC} = 10 V,	$I_{C(on)}$ = 2.5 mA,					50		μs
t_f	Fall Time	R_L = 100 Ω,	See Figure 1					50		

PARAMETER MEASUREMENT INFORMATION



TEST CIRCUIT



VOLTAGE WAVEFORMS

NOTES: a. The input waveform is supplied by a generator with the following characteristics: Z_{out} = 50 Ω, $t_r \leq$ 15 ns, duty cycle ≈ 1%, t_w = 100 μs.

b. The output waveform is monitored on an oscilloscope with the following characteristics: $t_r \leq$ 12 ns, $R_{in} \geq$ 1 MΩ, $C_{in} \leq$ 20 pF.

FIGURE 1–SWITCHING TIMES

TYPICAL CHARACTERISTICS

TIL113
COLLECTOR CURRENT
vs
COLLECTOR-EMITTER VOLTAGE

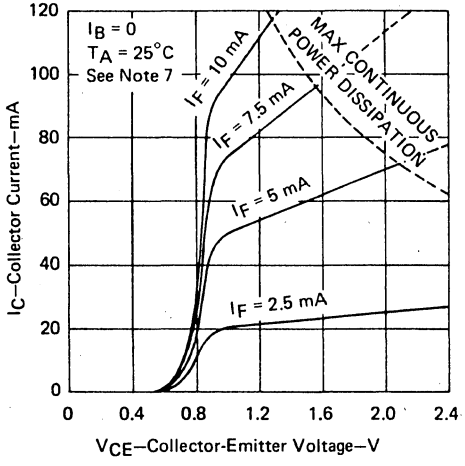


FIGURE 2

TIL119
COLLECTOR CURRENT
vs
COLLECTOR-EMITTER VOLTAGE

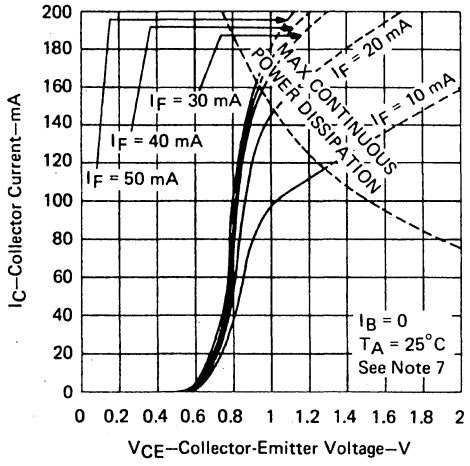


FIGURE 3

TIL113
COLLECTOR CURRENT
vs
INPUT-DIODE FORWARD CURRENT

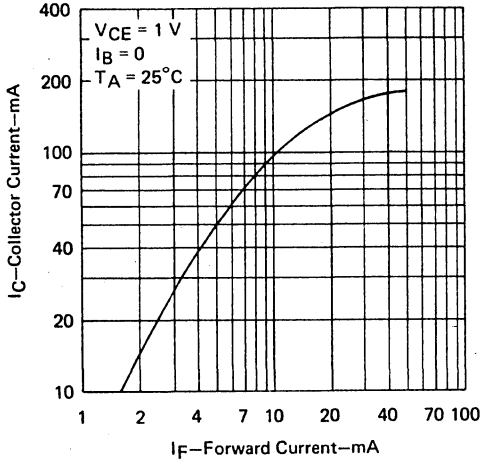


FIGURE 4

OFF-STATE COLLECTOR CURRENT
vs
FREE-AIR TEMPERATURE

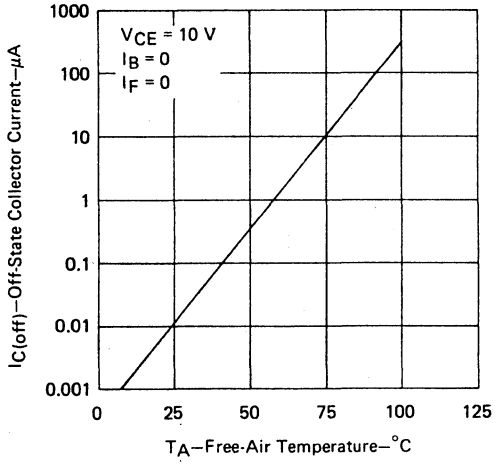


FIGURE 5

NOTE 7: Pulse operation of input diode is required for operation beyond limits shown by dotted line.

TYPICAL CHARACTERISTICS

TIL113
RELATIVE COLLECTOR-EMITTER SATURATION VOLTAGE
vs
FREE-AIR TEMPERATURE

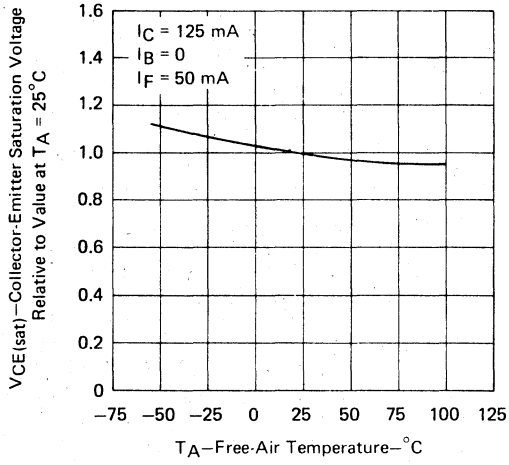


FIGURE 6

TIL113
TRANSISTOR STATIC FORWARD CURRENT TRANSFER RATIO
vs
COLLECTOR CURRENT

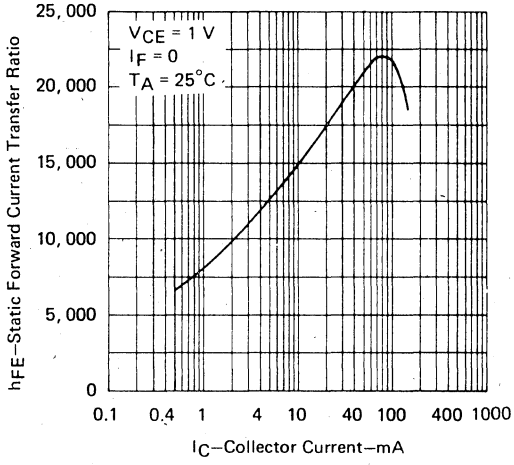


FIGURE 7

INPUT DIODE FORWARD CONDUCTION CHARACTERISTICS

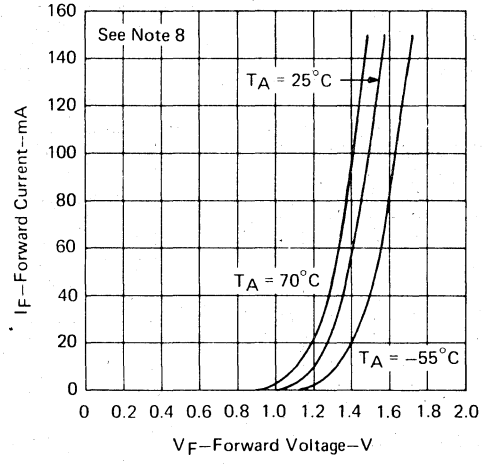


FIGURE 8

NOTE 8: This parameter was measured using pulse techniques. t_w = 1 ms, duty cycle ≤ 2%.

OPTO 22

I/O Module Detail
Electrical Specifications

AC INPUT MODULES	MODEL IAC5	MODEL IAC15	MODEL IAC24	MODEL IAC5-A	MODEL IAC15-A	MODEL IAC24-A
AC INPUT LINE VOLTAGE	95 to 130 VAC			180 to 280 VAC		
INPUT CURRENT AT RATED LINE	10 ma					
ISOLATION INPUT TO OUTPUT	2500 Volt RMS					
INPUT ALLOWED FOR NO OUTPUT	1.5 ma					
TURN ON TIME	20 Millisecond Maximum					
TURN OFF TIME	20 Millisecond Maximum					
OUTPUT TRANST. BREAKDOWN	30 Volts DC					
OUTPUT CURRENT	25 ma					
OUTPUT LEAKAGE 30VDC, NO. INPUT	100 Microamp Maximum					
OUTPUT VOLTAGE DROP	.4 Volts at 25 ma Load					
LOGIC SUPPLY VOLTAGE DC	4.5 to 6 V	12 to 18 V	20 to 30 V	4.5 to 6 V	12 to 18 V	20 to 30 V
LOGIC SUPPLY CURRENT	12 ma	15 ma	18 ma	12 ma	15 ma	18 ma

AC OUTPUT MODULES	MODEL OAC5	MODEL OAC15	MODEL OAC24	MODEL OAC5-A	MODEL OAC15-A	MODEL OAC24-A
LINE VOLTAGE	12 to 140 VAC			24 to 280 VAC		
CURRENT RATING	3 Amps①					
1-CYCLE SURGE	55 Amps Peak					
SIGNAL INPUT RESISTANCE	220 Ohm	1K Ohm	2.2K Ohm	220 Ohm	1K Ohm	2.2K Ohm
SIGNAL PICKUP VOLTS DC	3V 8V Ald.*	9V 16V Ald.*	18V 32V Ald.*	3V 8V Ald.*	9V 16V Ald.*	18V 32V Ald.*
SIGNAL DROPOUT VOLTS DC	1 Volt					
PEAK REPETITIVE VOLTAGE	400V			500 Volts		
MAXIMUM CONTACT DROP	1.6V					
OFF STATE LEAKAGE	5 ma RMS					
MINIMUM LOAD CURRENT	20 ma					
ISOLATION INPUT TO OUTPUT	2500 Volts RMS					
CAPACITANCE INPUT TO OUTPUT	8 Pf					
STATIC DV/DT	200 Volts/Microsecond Min					
COMMUTATING DV/DT	Built in snubber (will commutate .5 power factor loads)					

*Allowed

DC INPUT MODULES	MODEL IDC5	MODEL IDC15	MODEL IDC24
INPUT LINE VOLTAGE	10-32 VDC		
INPUT CURRENT	32 ma at 32V		
ISOLATION INPUT TO OUTPUT	2500 Volt RMS		
CAPACITANCE INPUT TO OUTPUT	8 Pf		
INPUT ALLOWED FOR NO OUTPUT	2 ma		
TURN ON TIME	5 Millisecond Max		
TURN OFF TIME	5 Millisecond Max		
OUTPUT TRANST. BREAKDOWN	30 Volts DC		
OUTPUT CURRENT	25 ma		
OUTPUT LEAKAGE 30 VDC NO INPUT	100 Microamps Max		
OUTPUT VOLTAGE DROP	.4 Volt at 25 ma		
LOGIC SUPPLY VOLTAGE	4.5 to 6V	12 to 18V	20 to 30V
LOGIC SUPPLY CURRENT	12 ma	15 ma	18 ma

DC OUTPUT MODULES	MODEL ODC5	MODEL ODC15	MODEL ODC24
LOAD VOLTAGE RATING	60V DC		
OUTPUT CURRENT RATING	3 Amps①		
OFF STATE LEAKAGE	1 ma Max		
ISOLATION INPUT TO OUTPUT	2500 V RMS		
SIGNAL PICK UP VOLTAGE	3V 8V Ald.*	9V 18V Ald.*	18V 28V Ald.*
SIGNAL DROP OUT VOLTAGE	1Volt		
SIGNAL INPUT RESISTANCE	220 Ohm	1K Ohm	2.2K Ohm
1 SECOND SURGE	5 Amps		
TURN ON TIME	500 Microsecond		
TURN OFF TIME	2.5 Millisecond		

*Allowed
①Derate .033 Amps per degree C from 20° C

opto 22

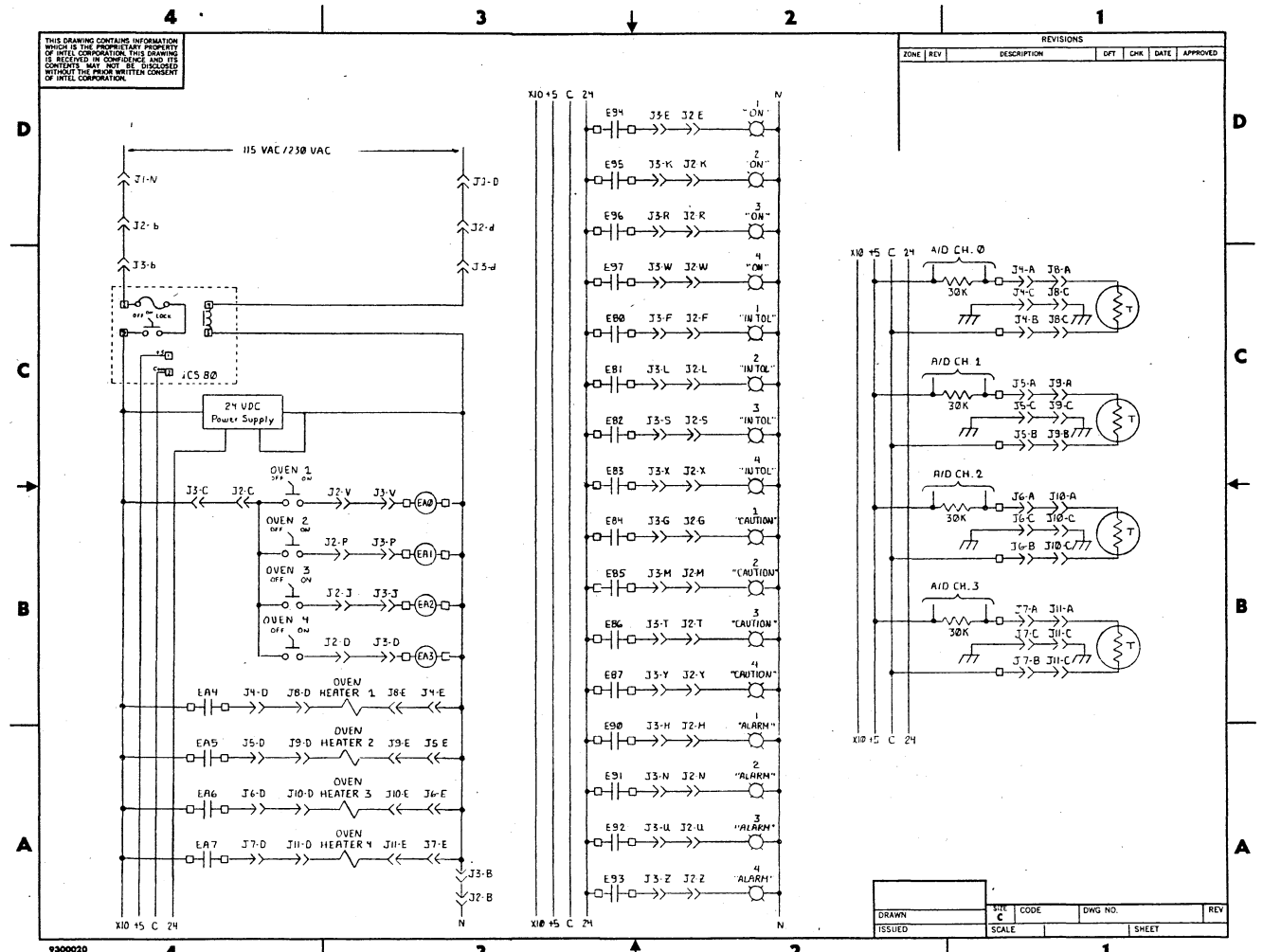
High Voltage DC Output Modules

DC OUTPUT MODULES	MODEL ODC5-A	MODEL ODC15-A	MODEL ODC24-A
LOAD VOLTAGE RATING	200V DC		
OUTPUT CURRENT RATING	1 Amps		
OFF STATE LEAKAGE	2 ma Max		
ISOLATION INPUT TO OUTPUT	2500 V RMS		
SIGNAL PICK UP VOLTAGE	3V 8V Ald.*	9V 18V Ald.*	18V 28V Ald.*
SIGNAL DROP OUT VOLTAGE	1Volt		
SIGNAL INPUT RESISTANCE	220 Ohm	1K Ohm	2.2K Ohm
1 SECOND SURGE	5 Amps		
TURN ON TIME	500 Microsecond		
TURN OFF TIME	2.5 Millisecond		

*Allowed

Fast Switching DC Input Modules

DC INPUT MODULES	MODEL IDC5-B	MODEL IDC15-B	MODEL IDC24-B
INPUT LINE VOLTAGE	4-16 VDC		
INPUT CURRENT	14 ma at 5V		
ISOLATION INPUT TO OUTPUT	2500 Volt RMS		
CAPACITANCE INPUT TO OUTPUT	8 Pf		
INPUT ALLOWED FOR NO OUTPUT	1 Volt		
TURN ON TIME	50 Microsecond Max		
TURN OFF TIME	100 Microsecond Max		
OUT TRANSISTOR BREAKDOWN	30 Volts DC		
OUTPUT CURRENT	25 ma		
OUTPUT LEAKAGE 30 VDC NO INPUT	100 Microamps Max		
OUTPUT VOLTAGE DROP	.4 Volt at 25 ma		
LOGIC SUPPLY VOLTAGE	4.5 to 6V	12 to 18V	20 to 30V
LOGIC SUPPLY CURRENT	12 ma		


THIS DRAWING CONTAINS INFORMATION WHICH IS THE PROPRIETARY PROPERTY OF INTEL CORPORATION. THIS DRAWING IS RECEIVED IN CONFIDENCE AND ITS CONTENTS MAY NOT BE DISCLOSED WITHOUT THE PRIOR WRITTEN CONSENT OF INTEL CORPORATION.
REVISIONS
ZONE | REV | DESCRIPTION | DFT | CHK | DATE | APPROVED
115 VAC / 230 VAC
J1-N
J2-b
J3-b
J1-D
J2-d
J3-d
OFF LOCK
1CS 8Ø
24 VDC Power Supply
OVEN 1
OVEN 2
OVEN 3
OVEN 4
J3-C J2-C J2-V J3-V EAØ
J2-P J3-P EA1
J2-J J3-J EA2
J2-D J3-D EA3
EA4 J4-D J8-D OVEN HEATER 1 J8-E J4-E
EA5 J5-D J9-D OVEN HEATER 2 J9-E J5-E
EA6 J6-D J10-D OVEN HEATER 3 J10-E J6-E
EA7 J7-D J11-D OVEN HEATER 4 J11-E J7-E
J3-B
J2-B
X10 +5 C 24
E94 J3-E J2-E "ON" 1
E95 J3-K J2-K "ON" 2
E96 J3-R J2-R "ON" 3
E97 J3-W J2-W "ON" 4
E80 J3-F J2-F "IN TOL" 1
E81 J3-L J2-L "IN TOL" 2
E82 J3-S J2-S "IN TOL" 3
E83 J3-X J2-X "IN TOL" 4
E84 J3-G J2-G "CAUTION" 1
E85 J3-M J2-M "CAUTION" 2
E86 J3-T J2-T "CAUTION" 3
E87 J3-Y J2-Y "CAUTION" 4
E90 J3-H J2-H "ALARM" 1
E91 J3-N J2-N "ALARM" 2
E92 J3-U J2-U "ALARM" 3
E93 J3-Z J2-Z "ALARM" 4
A/D CH. Ø
A/D CH. 1
A/D CH. 2
A/D CH. 3
30K
J4-A J8-A
J4-C J8-C
J4-B J8-C
J5-A J9-A
J5-C J9-C
J5-B J9-B
J6-A J10-A
J6-C J10-C
J6-B J10-C
J7-A J11-A
J7-C J11-C
J7-B J11-C
T
DRAWN
ISSUED
SIZE C
CODE
DWG NO.
REV
SCALE
SHEET
9300020

```
                $TITLE ('CONTROL TASK')
               /*******************************************************
               * This task handles the control and monitoring of      *
               * four oven chambers.                                  *
               *******************************************************/
1              CONTROLSTASK$MODULE:
               Do;
2     1        DECLARE EXCHANGE$DESCRIPTOR LITERALLY 'STRUCTURE (
                MESSAGE$HEAD ADDRESS,
                MESSAGE$TAIL ADDRESS,
                TASK$HEAD ADDRESS,
                TASK$TAIL ADDRESS,
                EXCHANGE$LINK ADDRESS)';
3     1        DECLARE TRUE LITERALLY '0FFH';
4     1        DECLARE FALSE LITERALLY '00H';
5     1        DECLARE BOOLEAN LITERALLY 'BYTE';
6     1        DECLARE FOREVER LITERALLY 'WHILE 1';
7     1        DECLARE MSG$HDR LITERALLY '
                LINK ADDRESS,
                LENGTH ADDRESS,
                TYPE BYTE,
                HOME$EX ADDRESS,
                RESP$EX ADDRESS';
8     1        DECLARE MSG$DESCRIPTOR LITERALLY 'STRUCTURE(
                MSG$HDR,
                REMAINDER(1) BYTE)';
               /* AIMSG.ELT - ANALOG INPUT REQUEST MESSAGE FORMAT */
9     1        DECLARE AIMSG LITERALLY 'STRUCTURE(
                              MSG$HDR,
                              STATUS ADDRESS,
                              BASE$PTR ADDRESS,
                              CHANNEL$GAIN ADDRESS,
                              ARRAY$PTR ADDRESS,
                              COUNT ADDRESS,
                              ACTUAL$COUNT ADDRESS)';
               /* AITYP.ELT - ANALOG INPUT MESSAGE TYPES */
10    1        DECLARE AIREP LITERALLY '30',
                       AISQS LITERALLY '31',
                       AISQV LITERALLY '32',
                       AITRN LITERALLY '33';
11    1          Declare  (n,k) byte;
12    1        Declare (MSG$PTR,LOCKOUT) address;
13    1          Declare (BLOCK0,BLOCK1,BLOCK2,BLOCK3) byte external;
14    1        Declare TOLERANCE(4) address external;
15    1        Declare TEMP(4) address external;
16    1        Declare SETPOINT(4) address external;
17    1        Declare T$AVERAGE(4) address;
18    1        Declare T$LAST(4) address;
19    1        Declare T$LAST$AVERAGE(4) address;
20    1        Declare T$t5(4) address;
21    1        Declare T$t10(4) address;
22    1        Declare STATUS(4) byte external;
23    1          Declare CRT$DISPLAY$LOCK(5) address external;
```

```
24   1       Declare TEMP$CALIBRATE(5) address external;
25   1       Declare DUMMY$EXCH(5) address external;
26   1       Declare TEMP$LOCKOUT$EXCH(5) address external;
27   1       Declare RQAIEX(5) address external;
28   1       Declare ASD$EXCH(5) address external;
29   1       Declare CONSTANT$LOCKOUT$EXCH(5) address external;
30   1       Declare CRT$STATUS$EXCH(5) address external;
31   1         Declare ALARM$MSG structure (MSG$HDR);
32   1       Declare CONVERT ai$msg;
             /* This term is used to convey initial temperatures */
33   1       Declare CAL$TEMP based MSG$PTR structure (
                 MSG$HDR,
                 CAL address );
34   1         RQWAIT:
                 Procedure (EXCH,MESSAGE) address external;
35   2           Declare (EXCH,MESSAGE) address;
36   2       end RQWAIT;
37   1         RQSEND:
                 Procedure (EXCH,MESSAGE) external;
38   2           Declare (EXCH,MESSAGE) address;
39   2       end RQSEND;
40   1         RQACPT:
                 Procedure (EXCH) address external;
41   2           Declare EXCH address;
42   2       end RQACPT;
43   1         Declare OVEN$IN$TOL(4) byte data (
                 01H,02H,04H,08H );
44   1       Declare OVEN$CAUTION(4) byte data (
                 10H,20H,40H,80H );
45   1       Declare OVEN$DANGER(4) byte data (
                 01H,02H,04H,08H );
46   1       Declare OVEN$ON$MASK(4) byte data (
                 01H,02H,04H,08H );
47   1       Declare OVEN$HEATER(4) byte data (
                 10H,20H,40H,80H );
48   1       Declare OVEN$RUN(4) byte data (
                 10H,20H,40H,80H );
49   1       Declare OFFSET(4) address;
50   1       Declare TABLE(255) address data (
                 200,201,202,203,204,205,206,207,208,209,
                 209,210,211,212,213,214,215,216,217,218,
                 219,220,221,222,223,224,225,226,227,228,
                 229,230,231,232,233,235,236,237,238,239,
                 240,241,243,244,245,247,248,249,250,251,
                 252,254,256,257,258,259,260,261,263,265,
                 266,267,268,269,270,271,273,274,276,278,
                 279,280,282,284,285,287,288,289,290,291,
                 293,295,296,298,299,300,302,304,305,307,
                 308,309,310,312,314,315,318,320,322,324,
                 326,328,330,332,334,336,338,340,342,344,
                 346,348,350,352,354,356,358,360,362,364,
                 366,368,370,372,374,376,378,380,382,385,
                 388,390,392,395,398,400,402,405,407,410,
                 412,415,418,420,423,426,428,430,433,436,
                 439,441,444,447,451,454,457,460,463,466,
```

```
                470,473,476,480,484,488,492,496,500,504,
                507,511,515,519,523,527,531,535,540,545,
                550,555,560,565,570,575,580,585,590,595,
                600,605,610,615,620,625,630,635,640,645,
                650,655,660,665,670,675,680,685,690,695,
                700,705,710,715,720,725,730,735,740,745,
                750,000,000,000,000,000,000,000,000,000,
                000,000,000,000,000,000,000,000,000,000,
                000,000,000,000,000,000,000,000,000,000,
                000,000,000,000,000,000                    );


                /* Initialization of control task */
51    1         CONTROL$TASK:
                Procedure public;
52    2           Output(235)=81H;
53    2           CONVERT.BASE$PTR=0F700H;
54    2           CONVERT.LENGTH=21;
55    2           CONVERT.TYPE=AISQS;
56    2           CONVERT.RESP$EX=.A$D$EXCH;
57    2           CONVERT.CHANNEL$GAIN=0;
58    2           CONVERT.ARRAY$PTR=.TEMP;
59    2           CONVERT.COUNT=4;
60    2           Do forever;
                  /* Wait for one second to elapse */
61    3             MSG$PTR=RQWAIT (.DUMMY$EXCH,20);
                  /* Bring in data from switches */
62    3             BLOCK0=NOT INPUT(224);
                  /* Lockout temperature storage areas for update */
63    3             LOCKOUT=RQWAIT (.TEMP$LOCKOUT$EXCH,0);
                  /* Get raw data from analog converter */
64    3             Call RQSEND (.RQAIEX,.CONVERT);
65    3             MSG$PTR=RQWAIT(.A$D$EXCH,0);
                  /* Temperature calibrate procedure */
66    3             MSG$PTR=RQACPT(.TEMP$CALIBRATE);
67    3             If MSG$PTR <> 0
                    then do;
69    4                 k=0;
70    4                 Do while (TABLE(k)<>CALTEMP.CAL AND
                                k<255);
71    5                   k=k+1;
72    5                 end;
73    4                 Do n=0 to 3;
74    5                   OFFSET(n)=(TEMP(n)/16)-k;
75    5                 end;
76    4             end;
                  /* Convert data into engineering units */
77    3             Do n=0 to 3;
78    4                 If ((TEMP(n)/16)-OFFSET(n))>255
                        then TEMP(n)=0;
80    4                 else TEMP(n)=TABLE((TEMP(n)/16)-OFFSET(n));
81    4             end;
                  /* Release lockout of temperatures */
82    3             Call RQSEND (.TEMP$LOCKOUT$EXCH,LOCKOUT);
                  /* Compute average temperature */
```

```
82   3          Do n=0 to 3;
84   4             T$AVERAGE(n)=(T$LAST(n)+TEMP(n))/2;
                /* Project temperatures into the future */
85   4             If T$AVERAGE(n)>=T$LAST$AVERAGE(n)
                    then do;
87   5               T$t5(n)=((T$AVERAGE(n)-T$LAST$AVERAGE(n))*5)
                          +T$LAST$AVERAGE(n);
88   5               T$t10(n)=((T$AVERAGE(n)-T$LAST$AVERAGE(n))*10)
                          +T$LAST$AVERAGE(n);
89   5             end;
90   4               else do;
91   5               T$t5(n)=T$LAST$AVERAGE(n)-((T$LAST$AVERAGE(n)
                          -T$AVERAGE(n))*5);
92   5               T$t10(n)=T$LAST$AVERAGE(n)-((T$LAST$AVERAGE(n)
                          -T$AVERAGE(n))*10);
93   5             end;
                /* Update stored data */
94   4             T$LAST$AVERAGE(n)=T$AVERAGE(n);
95   4             T$LAST(n)=TEMP(n);
                /* Test for active oven */
96   4             MSC$PTR=RQWAIT (.CONSTANT$LOCKOUT$EXCH,0);
97   4             If (((BLOCK0 AND OVEN$ON$MASK(n))<>0)
                    AND (TEMP(n)<>0))
                    then do;
99   5                 STATUS(n)=7;
100  5                 BLOCK2=BLOCK2 OR OVEN$RUN(n);
                /* Test for an intolerance condition */
101  5                 If SETPOINT(n)-TOLERANCE(n) < TEMP(n) AND
                         SETPOINT(n)+TOLERANCE(n) > TEMP(n)
                      then do;
103  6                     STATUS(n)=7;
104  6                     BLOCK1=BLOCK1 OR OVEN$IN$TOL(n);
105  6                   end;
106  5                   else BLOCK1=BLOCK1 AND NOT OVEN$IN$TOL(n);
                /* Test for a caution condition */
107  5                 If SETPOINT(n)-TOLERANCE(n) > T$t5(n) OR
                         SETPOINT(n)+TOLERANCE(n) < T$t5(n)
                      then do;
109  6                     STATUS(n)=14;
110  6                     BLOCK1=BLOCK1 OR OVEN$CAUTION(n);
111  6                   end;
112  5                   else BLOCK1=BLOCK1 AND NOT OVEN$CAUTION(n);
                /* Test for a danger condition */
113  5                 If SETPOINT(n)-TOLERANCE(n) > TEMP(n) OR
                         SETPOINT(n)+TOLERANCE(n) < TEMP(n)
                      then do;
115  6                     STATUS(n)=21;
116  6                     BLOCK2=BLOCK2 OR OVEN$DANGER(n);
117  6                   end;
118  5                   else BLOCK2=BLOCK2 AND NOT OVEN$DANGER(n);
                /* Handle control of heater elements */
119  5                 If SETPOINT(n) > T$t10(n)
                      then BLOCK3=BLOCK3 OR OVEN$HEATER(n);
121  5                 else BLOCK3=BLOCK3 AND NOT OVEN$HEATER(n);
122  5               end;
123  4             else do;
                /* Turn everything off when operator shuts off oven */
124  5                 BLOCK1=BLOCK1 AND NOT OVEN$IN$TOL(n);
125  5                 BLOCK1=BLOCK1 AND NOT OVEN$CAUTION(n);
126  5                 BLOCK3=BLOCK3 AND NOT OVEN$HEATER(n);
```

```
127   5                    BLOCK2=BLOCK2 AND NOT OVEN$DANGER(n);
128   5                    BLOCK2=BLOCK2 AND NOT OVEN$RUN(n);
129   5                    STATUS(n)=0;
130   5                end;
131   4                Call RQSEND(.CONSTANT$LOCKOUT$EXCH,MSG$PTR);
132   4            end;

                   /* Output data to real world */
133   3            OUTPUT(232)=BLOCK1;
134   3            OUTPUT(233)=BLOCK2;
135   3            OUTPUT(234)=BLOCK3;
136   3        end;
137   2        end CONTROL$TASK;
138   1        end CONTROL$TASK$MODULE;

                 MODULE INFORMATION:

                      CODE AREA SIZE     = 0946H   2374D
                      VARIABLE AREA SIZE = 0054H     84D
                      MAXIMUM STACK SIZE = 0006H      6D
                      235 LINES READ
                      0 PROGRAM ERROR(S)

                 END OF PL/M-80 COMPILATION


                 $TITLE('CRT PARAMETER TASK')
                 /*********************************************
                 * This task is used to examine and update the *
                 * temperature setpoints and tolerances for    *
                 * each of the four ovens.                     *
                 *********************************************/
   1             UPDATE$TASK:
                 Do;

                 $Include (:F0:COMMON.ELT)
   2   1   =     DECLARE TRUE LITERALLY '0FFH';
   3   1   =     DECLARE FALSE LITERALLY '00H';
   4   1   =     DECLARE BOOLEAN LITERALLY 'BYTE';
   5   1   =     DECLARE FOREVER LITERALLY 'WHILE 1';
                 $Include (:F0:MSGTYP.ELT)
   6   1   =     DECLARE DATA$TYPE LITERALLY '0',
           =             INT$TYPE LITERALLY '1',
           =             MISSED$INT$TYPE LITERALLY '2',
           =             TIME$OUT$TYPE LITERALLY '3',
           =             FS$REQ$TYPE LITERALLY '4',
           =             UC$REQ$TYPE LITERALLY '5',
           =             FS$NAK$TYPE LITERALLY '6',
           =             CNTRL$C$TYPE LITERALLY '7',
           =             READ$TYPE LITERALLY '8',
           =             CLR$RD$TYPE LITERALLY '9',
           =             LAST$RD$TYPE LITERALLY '10',
           =             ALARM$TYPE LITERALLY '11',
           =             WRITE$TYPE LITERALLY '12';
                 $Include (:F0:MSG.ELT)
```

```
 7    1  =    DECLARE MSG$HDR LITERALLY '
          =     LINK ADDRESS,
          =     LENGTH ADDRESS,
          =     TYPE BYTE,
          =     HOME$EX ADDRESS,
          =     RESP$EX ADDRESS';
          =
 8    1  =    DECLARE MSG$DESCRIPTOR LITERALLY 'STRUCTURE(
          =     MSG$HDR,
          =     REMAINDER(1) BYTE)';
               $Include (:F0:THMSG.ELT)
 9    1  =    DECLARE TH$MSG LITERALLY 'STRUCTURE (
          =     MSGHDR,
          =     STATUS ADDRESS,
          =     BUFFER$ADR ADDRESS,
          =     COUNT ADDRESS,
          =     ACTUAL ADDRESS,
          =     REMAINDER(120) BYTE)';
10    1  =    DECLARE MIN$TH$MSG$LENGTH LITERALLY '17';
               $Include (:F0:CHAR.ELT)
          =
          =    /* SPECIAL ASCII CHARACTERS */
          =
11    1  =    DECLARE
          =      NULL             LITERALLY '00H',
          =      CONTROL$C        LITERALLY '03H',
          =      CONTROL$E        LITERALLY '05H',
          =      BELL             LITERALLY '07H',
          =      TAB              LITERALLY '09H',
          =      LF               LITERALLY '0AH',
          =      VT               LITERALLY '0BH',
          =      FF               LITERALLY '0CH',
          =      CR               LITERALLY '0DH',
          =      CONTROL$P        LITERALLY '10H',
          =      CONTROL$Q        LITERALLY '11H',
          =      CONTROL$R        LITERALLY '12H',
          =      CONTROL$S        LITERALLY '13H',
          =      CONTROL$X        LITERALLY '18H',
          =      CONTROL$Z        LITERALLY '1AH',
          =      ESC              LITERALLY '1BH',
          =      QUOTE            LITERALLY '22H',
          =      LCA              LITERALLY '61H',
          =      LCZ              LITERALLY '7AH',
          =      RUBOUT           LITERALLY '7FH';
          =
               $Include (:F0:SYNCH.EXT)
12    1  =    RQSEND:
          =      PROCEDURE (EXCHANGE$POINTER,MESSAGE$POINTER) EXTERNAL;
13    2  =        DECLARE (EXCHANGE$POINTER,MESSAGE$POINTER) ADDRESS;
          =
14    2  =      END RQSEND;
          =
15    1  =    RQWAIT:
          =      PROCEDURE (EXCHANGE$POINTER,DELAY) ADDRESS EXTERNAL;
16    2  =        DECLARE (EXCHANGE$POINTER,DELAY) ADDRESS;
          =
```

```
17   2  =      END RQWAIT;
        =
18   1  =    RQACPT:
        =      PROCEDURE (EXCHANGE$POINTER) ADDRESS EXTERNAL;
19   2  =        DECLARE EXCHANGE$POINTER ADDRESS;
        =
20   2  =      END RQACPT;
        =
21   1  =    RQISND:
        =      PROCEDURE (IED$PTR) EXTERNAL;
22   2  =        DECLARE IED$PTR ADDRESS;
        =
23   2  =      END RQISND;
24   1        Declare TEMP$CALIBRATE(5) address external;
25   1        Declare UPDATE$EXCH(5) address external;
26   1        Declare CRT$STATUS$EXCH(5) address external;
27   1        Declare COMP$EXCH(5) address external;
28   1        Declare CONSTANT$LOCKOUT$EXCH(5) address external;
29   1        Declare RQOUTX(5) address external;
30   1        Declare RQINPX(5) address external;
31   1        Declare WORDS$EXCH(5) address external;
32   1        Declare SETPOINT(4) address external;
33   1        Declare TOLERANCE(4) address external;
34   1        Declare BUFFER2 address;
35   1        Declare MSG$PTR address;
36   1        Declare MSG structure (
                 MSG$HDR,
                 STATUS address,
                 BUFFER$PTR address,
                 COUNT address,
                 ACTUAL address );
37   1        Declare CAL$TEMP structure (
                 MSG$HDR,
                 CAL address );
38   1        Declare UPD$MSG address;
39   1        Declare ENERGIZE based UPD$MSG structure (
                 MSG$HDR,
                 STATUS address,
                 BUFFER$PTR address,
                 COUNT address,
                 ACTUAL address );
40   1        Declare ENABLE$MSG structure (
                 MSG$HDR );
41   1        Declare BUFFER(80) byte;
42   1        Declare OVEN byte;
43   1          DEC$REP:
                   Procedure (SOURCE,TARGET) external;
44   2             Declare (SOURCE,TARGET) address;
45   2          end DEC$REP;
46   1          ASC$2$BINARY:
                     Procedure (SOURCE,TARGET,SIZE) byte external;
47   2             Declare (SOURCE,TARGET) address;
48   2             Declare SIZE byte;
49   2          end ASC$2$BINARY;
50   1          Declare MSG$1(20) byte data (
                   ESC,'E','ENTER OVEN NUMBER-');
```

```
51   1       Declare MSG$2(28) byte data (
                 CR,LF,
                 'ENTER NEW SETPOINT-',
                 'XXXX.X-' );
52   1       Declare MSG$3(29) byte data (
                 CR,LF,
                 'ENTER NEW TOLERANCE-',
                 'XXXX.X-' );
53   1       Declare CALMSG(12) byte data (
                 'TEMPERATURE-' );
54   1       Declare MSG$4(62) byte data (
                 CR,LF,
                 '(STATUS-(S), PARAMETERS-(P), CALIBRATE-(C))',
                 CR,LF,
                 'ENTER REQUEST-' );
55   1       Declare WAIT literally 'MSG$PTR=';
56   1       Declare FOR literally 'RQWAIT';
57   1       Declare START literally 'CALL';
58   1       Declare TASK literally 'RQSEND';
59   1     UPDATE:
       Procedure public;
         /* Initialize task at start-up time */
60   2       Do forever;
61   3       MSG.RESP$EX=.COMP$EXCH;
         /* Wait for request to enter task */
62   3            UPD$MSG=RQWAIT (.UPDATE$EXCH,0);
         /* Get desired oven number from operator */
63   3            RQST$OVEN:
                  MSG.BUFFER$PTR=.MSG$1;
64   3            MSG.TYPE=WRITE$TYPE;
65   3            MSG.COUNT=20;
66   3            Start task (.RQOUTX,.MSG);
67   3            Wait for (.COMP$EXCH,0);
         /* ...Input new number */
68   3            MSG.BUFFER$PTR=.BUFFER;
69   3            MSG.COUNT=255;
70   3            MSG.TYPE=CLR$RD$TYPE;
71   3            Start task (.RQINPX,.MSG);
72   3            Wait for (.COMP$EXCH,0);
73   3            OVEN=(BUFFER(0) AND 07E)-1;
74   3            If OVEN >3 then go$to RQST$OVEN;
         /* Display request and current setpoint */
76   3            GET$TEMP:
                  Call move (28,.MSG$2,.BUFFER);
77   3            Call DEC$REP (.SETPOINT(oven),.BUFFER+21);
78   3              MSG.TYPE=WRITE$TYPE;
79   3            MSG.COUNT=28;
80   3            Start task (.RQOUTX,.MSG);
81   3            Wait for (.COMP$EXCH,0);
         /* ... Input new setpoint */
82   3            MSG.TYPE=CLR$RD$TYPE;
83   3            Start task (.RQINPX,.MSG);
84   3            Wait for (.COMP$EXCH,0);
85   3            If ASC$2$BINARY(.BUFFER,.BUFFER2,1)=0 OR BUFFER2 > 700
                  then go$to GET$TEMP;
87   3            If BUFFER2 <> 0
                  then do;
89   4                Wait for (.CONSTANT$LOCKOUT$EXCH,0);
90   4                SETPOINT(oven)=BUFFER2;
91   4                Start task (.CONSTANT$LOCKOUT$EXCH,MSG$PTR);
```

```
 92   4          end;
           /* Display request and current tolerance */
 93   3          GET$TOL:
                 Call move (29,.MSG$3,.BUFFER);
 94   3          Call DEC$REP (.TOLERANCE(oven),.BUFFER+22);
 95   3          MSG.TYPE=WRITE$TYPE;
 96   3          MSG.COUNT=29;
 97   3          Start task (.RQOUTX,.MSG);
 98   3          Wait for (.COMP$EXCH,0);
           /* ...Input new tolerance */
 99   3          MSG.TYPE=CLR$RD$TYPE;
100   3          Start task (.RQINPX,.MSG);
101   3          Wait for (.COMP$EXCH,0);
102   3          If ASC$2$BINARY(.BUFFER,.BUFFER2,1)=0 OR BUFFER2 > 700
                 then go$to GET$TOL;
104   3          If BUFFER2 <> 0
                 then do;
106   4              Wait for (.CONSTANT$LOCKOUT$EXCH,0);
107   4              TOLERANCE(oven)=BUFFER2;
108   4              Start task (.CONSTANT$LOCKOUT$EXCH,MSG$PTR);
109   4          end;
           /* Ask operator if he is finished */
110   3          REQ$NEXT:
                 MSG.TYPE=WRITE$TYPE;
111   3          MSG.COUNT=52;
112   3          MSG.BUFFER$PTR=.MSG$4;
113   3          Start task (.RQOUTX,.MSG);
114   3          Wait for (.COMP$EXCH,0);
           /* ...Get his response */
115   3          MSG.TYPE=CLR$RD$TYPE;
116   3          MSG.BUFFER$PTR=.BUFFER;
117   3          Start task (.RQINPX,.MSG);
118   3          Wait for (.COMP$EXCH,0);
119   3          If (BUFFER(0) <>'S' AND BUFFER(0) <>'P'
                  AND BUFFER(0) <> 'C')
                 then go$to REQ$NEXT;
121   3          If BUFFER(0)='P'
                   then go$to RQST$OVEN;
123   3          If BUFFER(0)='C'
                 then do;
125   4              GET$CAL:
                     MSG.TYPE=WRITE$TYPE;
126   4              MSG.COUNT=12;
127   4              MSG.BUFFER$PTR=.CALMSG;
128   4              Start task (.RQOUTX,.MSG);
129   4              Wait for (.COMP$EXCH,0);
130   4              MSG.TYPE=CLR$RD$TYPE;
131   4              MSG.BUFFER$PTR=.BUFFER;
132   4              Start task (.RQINPX,.MSG);
133   4              Wait for (.COMP$EXCH,0);
134   4              If ASC$2$BINARY(.BUFFER,.BUFFER2,1) =0
                       OR BUFFER2>350 OR BUFFER2<200
                     then go$to GET$CAL;
136   4              CAL$TEMP.CAL=BUFFER2;
137   4              Call RQSEND (.TEMP$CALIBRATE,.CAL$TEMP);
138   4          end;
```

```
                MODULE INFORMATION:
                     CODE AREA SIZE      = 03C3H     963D
                     VARIABLE AREA SIZE  = 007CH     124D
                     MAXIMUM STACK SIZE  = 0004H       4D
                     264 LINES READ
                     0 PROGRAM ERROR(S)
                END OF PL/M-80 COMPILATION

139    3          ENERGIZE.TYPE=100;
140    3          Start task (.CRT$STATUS$EXCH,UPD$MSG);

141    3       end;
142    2    end UPDATE;
143    1 end UPDATE$TASK;



                $TITLE('CRT UPDATE TASK')
                /************************************************
                * This task is utilized to update the CRT ter-  *
                * minal display with the current operating par- *
                * ameters. It will be entered upon sytem start- *
                * up, upon operator request, or when a problem  *
                * exists with any of the activated ovens.       *
                ************************************************/
 1              CRT$DATA$MODULE:
                Do;
                $INCLUDE(:F0:SYNCH.EXT)
 2    1    =    RQSEND:
      =         PROCEDURE (EXCHANGE$POINTER,MESSAGE$POINTER) EXTERNAL;
 3    2    =      DECLARE (EXCHANGE$POINTER,MESSAGE$POINTER) ADDRESS;
      =
 4    2    =      END RQSEND;
      =
 5    1    =    RQWAIT:
      =         PROCEDURE (EXCHANGE$POINTER,DELAY) ADDRESS EXTERNAL;
 6    2    =      DECLARE (EXCHANGE$POINTER,DELAY) ADDRESS;
      =
 7    2    =      END RQWAIT;
      =
 8    1    =    RQACPT:
      =         PROCEDURE (EXCHANGE$POINTER) ADDRESS EXTERNAL;
 9    2    =      DECLARE EXCHANGE$POINTER ADDRESS;
      =
10    2    =      END RQACPT;
      =
11    1    =    RQISND:
      =         PROCEDURE (IED$PTR) EXTERNAL;
12    2    =      DECLARE IED$PTR ADDRESS;
      =
13    2    =      END RQISND;
                $INCLUDE (:F0:MSGTYP.ELT)
14    1    =    DECLARE DATA$TYPE LITERALLY '0',
```

```
          =           INT$TYPE LITERALLY '1',
          =           MISSED$INT$TYPE LITERALLY '2',
          =           TIME$OUT$TYPE LITERALLY '3',
          =           FS$REQ$TYPE LITERALLY '4',
          =           UC$REQ$TYPE LITERALLY '5',
          =           FS$NAK$TYPE LITERALLY '6',
          =           CNTRL$C$TYPE LITERALLY '7',
          =           READ$TYPE LITERALLY '8',
          =           CLR$RD$TYPE LITERALLY '9',
          =           LAST$RD$TYPE LITERALLY '10',
          =           ALARM$TYPE LITERALLY '11',
          =           WRITE$TYPE LITERALLY '12';
              $INCLUDE (:F0:EXCH.ELT)
15    1   =   DECLARE EXCHANGE$DESCRIPTOR LITERALLY 'STRUCTURE (
          =    MESSAGE$HEAD ADDRESS,
          =    MESSAGE$TAIL ADDRESS,
          =    TASK$HEAD ADDRESS,
          =    TASK$TAIL ADDRESS,
          =    EXCHANGE$LINK ADDRESS)';
              $INCLUDE (:F0:COMMON.ELT)
16    1   =   DECLARE TRUE LITERALLY '0FFH';
17    1   =   DECLARE FALSE LITERALLY '00H';
18    1   =   DECLARE BOOLEAN LITERALLY 'BYTE';
19    1   =   DECLARE FOREVER LITERALLY 'WHILE 1';
              $INCLUDE (:F0:MSG.ELT)
20    1   =   DECLARE MSG$HDR LITERALLY '
          =    LINK ADDRESS,
          =    LENGTH ADDRESS,
          =    TYPE BYTE,
          =    HOME$EX ADDRESS,
          =    RESP$EX ADDRESS';
          =
21    1   =   DECLARE MSG$DESCRIPTOR LITERALLY 'STRUCTURE(
          =    MSG$HDR,
          =    REMAINDER(1) BYTE)';
              $INCLUDE (:F0:CHAR.ELT)
          =
          =   /* SPECIAL ASCII CHARACTERS */
          =
22    1   =   DECLARE
          =     NULL            LITERALLY '00H',
          =     CONTROL$C       LITERALLY '03H',
          =     CONTROL$E       LITERALLY '05H',
          =     BELL            LITERALLY '07H',
          =     TAB             LITERALLY '09H',
          =     LF              LITERALLY '0AH',
          =     VT              LITERALLY '0BH',
          =     FF              LITERALLY '0CH',
          =     CR              LITERALLY '0DH',
          =     CONTROL$P       LITERALLY '10H',
          =     CONTROL$Q       LITERALLY '11H',
          =     CONTROL$R       LITERALLY '12H',
          =     CONTROL$S       LITERALLY '13H',
          =     CONTROL$X       LITERALLY '18H',
          =     CONTROL$Z       LITERALLY '1AH',
          =     ESC             LITERALLY '1BH',
          =     QUOTE           LITERALLY '22H',
```

```
          =     LCA             LITERALLY '61H',
          =     LCZ             LITERALLY '7AH',
          =     RUBOUT          LITERALLY '7FH';
          =
              $INCLUDE (:F?:THMSG.ELT)
23    1   =   DECLARE TH$MSG LITERALLY 'STRUCTURE (
          =    MSGHDR,
          =    STATUS ADDRESS,
          =    BUFFER$ADR ADDRESS,
          =    COUNT ADDRESS,
          =    ACTUAL ADDRESS,
          =    REMAINDER(128) BYTE)';
24    1   =   DECLARE MIN$TH$MSG$LENGTH LITERALLY '17';
25    1       Declare HOME literally '1BH,48H';
26    1       Declare L1$IMAGE(90) byte data (
                Home,Lf,Lf,Lf,Lf,Lf,
                'TEMPERATURE      ',
                '                 ',
                '                 ',
                '                 ',
                '                   ',
                'DEGREES C.' );
27    1     Declare L2$IMAGE(92) byte data (
                Home,Lf,Lf,Lf,Lf,Lf,Lf,Lf,
                'SETPOINT         ',
                '                 ',
                '                 ',
                '                 ',
                '                   ',
                'DEGREES C.' );
28    1     Declare L3$IMAGE(94) byte data (
                Home,Lf,Lf,Lf,Lf,Lf,Lf,Lf,Lf,Lf,
                'TOLERANCE        ',
                '                 ',
                '                 ',
                '                 ',
                '                   ',
                'DEGREES C.' );
29    1     Declare L4$IMAGE(75) byte data (
                Home,Lf,Lf,Lf,Lf,Lf,Lf,Lf,Lf,Lf,Lf,Lf,
                'STATUS           ',
                '  OFF            ',
                '  OFF            ',
                '  OFF            ',
                '  OFF   ' );
30    1     Declare CRT$HDR(168) byte data (
                1BH,45H,'                                        ',
                'OVEN STATUS DISPLAY',
                Cr,Lf,Lf,'                    ',
                  'OVEN-1         ',
                'OVEN-2           ',
                'OVEN-3           ',
                'OVEN-4',
```

```
                Cr,Lf,Lf,Lf,Lf,Lf,Lf,Lf,Lf,Lf,Lf,Lf,Lf,Lf,Lf,Lf,Lf,Lf,Lf,Lf,
            Lf,
                'TYPE ESCAPE TO ADJUST SETPOINTS' );
31    1     Declare BELLS(4) byte data (
                Bell,Bell,Bell,Bell );
32    1     Declare MESSAGES(35) byte data (
                '  OFF  ',
                '   OK  ',
                'CAUTION',
                ' ALARM ',
                '       ' );
33    1       Declare DISPLAY$PTR1(4) address data (
                .WORK$BUFF+23,
                .WORK$BUFF+36,
                .WORK$BUFF+49,
                .WORK$BUFF+62 );
34    1     Declare DISPLAY$PTR2(4) address data (
                .WORK$BUFF+25,
                .WORK$BUFF+38,
                .WORK$BUFF+51,
                .WORK$BUFF+64 );
35    1     Declare DISPLAY$PTR3(4) address data (
                .WORK$BUFF+27,
                .WORK$BUFF+40,
                .WORK$BUFF+53,
                .WORK$BUFF+66 );
36    1     Declare DISPLAY$PTR4(4) address data (
                .WORKBUFF+30,
                .WORKBUFF+43,
                .WORKBUFF+56,
                .WORKBUFF+69 );
37    1     Declare MSG$PTR address;
38    1     Declare MSG based MSG$PTR structure (
                MSG$HDR,
                COUNT address );
39    1     Declare STARTER(3) structure (
                MSG$HDR );
40    1     Declare READ structure (
                MSG$HDR,
                STATUS address,
                BUFFER$PTR address,
                COUNT address,
                ACTUAL address );
41    1     Declare DISPLAY$TEMP(4) structure (
                UPPER address,
                LOWER address );
42    1     Declare DISPLAY$SET(4) structure (
                LOWER address,
                UPPER address );
43    1     Declare DISPLAY$TOL(4) structure (
                LOWER address,
                UPPER address );
```

```
44   1          Declare OVEN$ON(4) byte data (
                   01H,02H,04H,08H );
45   1        Declare OVEN$CAUTION(4) byte data (
                   10H,20H,40H,80H );
46   1          Declare CRT structure (
                   MSG$HDR,
                   STATUS address,
                   BUFFER$PTR address,
                   COUNT address,
                   ACTUAL address );
47   1          Declare CRTLOCK structure (MSG$HDR);
48   1          Declare CRT$DISPLAY$LOCK(5) address external;
49   1        Declare TEMP$LOCKOUT$EXCH(5) address external;
50   1        Declare CONSTANT$LOCKOUT$EXCH(5) address external;
51   1        Declare CRT$EXCH(5) address external;
52   1        Declare CRT$STATUS$EXCH(5) address external;
53   1        Declare DUMMY$EXCH(5) address external;
54   1        Declare READ$BUFFER$EXCH(5) address external;
55   1        Declare UPDATE$EXCH(5) address external;
56   1        Declare RQINPX(5) address external;
57   1        Declare RQOUTX(5) address external;
58   1        Declare RQWAKE(5) address external;
59   1        Declare RQL7EX(5) address external;
60   1        Declare RQL6EX(5) address external;
61   1        Declare RQDBUC(5) address external;
62   1        Declare RQALRM(5) address external;
63   1          Declare TEMP(4) address external;
64   1        Declare DISP$TEMP(4) address;
65   1        Declare SETPOINT(4) address external;
66   1        Declare DISP$SETPNT(4) address;
67   1        Declare TOLERANCE(4) address external;
68   1        Declare DISP$TOL(4) address;
69   1        Declare STATUS(4) byte external;
70   1        Declare DISP$STAT(4) byte;
71   1        Declare (BLOCK1,BLOCK2) byte external;
72   1        Declare WORK$BUFF(170) byte;
73   1        Declare BUFFER$A(70) byte;
74   1        Declare (CHANGE,n,ALARM,NEW,BLANKER) byte;
75   1          Declare START literally 'call';
76   1        Declare TASK literally 'rqsend';
77   1        Declare WAIT literally 'msg$ptr=';
78   1        Declare For literally 'rqwait';
79   1          DEC$REP:
                   Procedure(SOURCE,TARGET) external;
80   2              Declare (SOURCE,TARGET) address;
81   2        end DEC$REP;
```

```
 82   1      CRT$DATA$TASK:
             Procedure public;
             /* Initialize system at start-up time */
 83   2        Start task (.TEMP$LOCKOUT$EXCH,.STARTER(0));
 84   2        Start task (.CONSTANT$LOCKOUT$EXCH,.STARTER(1));
 85   2        STARTER(2).TYPE=100;
 86   2        Start task (.CRT$STATUS$EXCH,.STARTER(2));
 87   2        CRT.RESP$EX=.CRT$EXCH;
             /* Perform main CRT wait */
 88   2        Do forever;
 89   3            Wait for (.DUMMY$EXCH,10);
 90   3            Wait for (.CRT$STATUS$EXCH,0);
 91   3            If MSG.TYPE=255
                  then ALARM=1;
 93   3            else ALARM=0;
             /* Output heading */
 94   3            If (MSG.TYPE=100 OR MSG.TYPE=255)
                  then do;
 96   4                If ALARM=0
                        then call RQSEND(.CRT$DISPLAY$LOCK,.CRTLOCK);
 98   4                CRT.TYPE=WRITE$TYPE;
 99   4                CRT.COUNT=167;
100   4                CRT.BUFFER$PTR=.WORK$BUFF;
101   4                READ.TYPE=CLR$RD$TYPE;
102   4                READ.COUNT=255;
103   4                READ.RESP$EX=.READ$BUFFER$EXCH;
104   4                READ.BUFFER$PTR=.BUFFERA;
105   4                If ALARM=0
                       then start task (.RQINPX,.READ);
107   4                Call move (82,.CRT$HDR,.WORK$BUFF);
108   4                Call move (86,.CRT$HDR+82,.WORK$BUFF+82);
109   4                Start task (.RQOUTX,.CRT);
110   4                Wait for (.CRT$EXCH,0);
111   4                NEW=1;
112   4            end;
             /* Test for change in temperature of any oven */
113   3            CHANGE=0;
114   3            Wait for (.TEMP$LOCKOUT$EXCH,0);
115   3            Do n=0 to 3;
116   4                If TEMP(n)<>DISP$TEMP(n)
                       then CHANGE=1;
118   4            end;
119   3            Call move (8,.TEMP,.DISP$TEMP);
120   3            Start task (.TEMP$LOCKOUT$EXCH,MSG$PTR);
             /* When a change exists build new line */
121   3            If CHANGE OR NEW
                  then do;
123   4                Call move (90,.L1$IMAGE,.WORK$BUFF);
124   4                Do n=0 to 3;
125   5                    Call DEC$REP(.DISP$TEMP(n),DISPLAY$PTR1(n));
126   5                end;
             /* Output new temperature line to CRT */
127   4                  CRT.TYPE=WRITE$TYPE;
128   4                CRT.COUNT=87;
```

```
129   4             Start task (.RQOUTX,.CRT);
130   4             Wait for (.CRT$EXCH,0);
131   4         end;
          /* Test for change in oven setpoints */
132   3         CHANGE=0;
133   3         Wait for (.CONSTANT$LOCKOUT$EXCH,0);
134   3         Do n=0 to 3;
135   4             If SETPOINT(n)<>DISP$SETPNT(n)
                  then CHANGE=1;
137   4         end;
138   3         Call move (8,.SETPOINT,.DISP$SETPNT);
139   3         Start task (.CONSTANT$LOCKOUT$EXCH,MSG$PTR);
          /* Build new line when a change was detected */
140   3         If CHANGE OR NEW
                  then do;
142   4             Call move (92,.L2$IMAGE,.WORKBUFF);
143   4             Do n=0 to 3;
144   5                 Call DEC$REP(.DISP$SETPNT(n),DISPLAY$PTR2(n));
145   5             end;
          /* Output setpoint line */
146   4             CRT.TYPE=WRITE$TYPE;
147   4             CRT.COUNT=89;
148   4             CRT.BUFFER$PTR=.WORKBUFF;
149   4             Start task (.RQOUTX,.CRT);
150   4             Wait for (.CRT$EXCH,0);
151   4         end;
          /* Test for change in tolerance line */
152   3         CHANGE=0;
153   3         Wait for (.CONSTANT$LOCKOUT$EXCH,0);
154   3         Do n=0 to 3;
155   4             If TOLERANCE(n)<>DISP$TOL(n)
                  then CHANGE=1;
157   4         end;
158   3         Call move (8,.TOLERANCE,.DISP$TOL);
159   3         Start task (.CONSTANT$LOCKOUT$EXCH,MSG$PTR);
          /* When change is found, build new line */
160   3         If CHANGE OR NEW
                  then do;
162   4             Call move (94,.L3$IMAGE,.WORK$BUFF);
163   4             Do n=0 to 3;
164   5                 Call DEC$REP(.DISP$TOL(n),DISPLAY$PTR3(n));
165   5             end;
          /* Output tolerance line */
166   4             CRT.TYPE=WRITE$TYPE;
167   4             CRT.COUNT=91;
168   4             CRT.BUFFER$PTR=.WORKBUFF;
169   4             Start task (.RQOUTX,.CRT);
170   4             Wait for (.CRT$EXCH,0);
171   4         end;
            /* Build status message */
172   3         CHANGE=0;
173   3         Wait for (.CONSTANT$LOCKOUT$EXCH,0);
174   3         Do n=0 to 3;
175   4             If STATUS(n)<>DISP$STAT(n)
                  then CHANGE=1;
```

```
177   4           end;
178   3           Call move (4,.STATUS,.DISP$STAT);
179   3           Start task (.CONSTANT$LOCKOUT$EXCH,MSG$PTR);
          /* Output to display */
180   3           If CHANGE OR NEW
                  then do;
182   4               Call move (75,.L4IMAGE,.WORK$BUFF);
183   4               Do n=0 to 3;
184   5                   Call move(7,.MESSAGES+DISP$STAT(n),DISPLAY$PTR4(
          n));
185   5               end;
186   4               CRT.COUNT=76;
187   4               Start task (.RQOUTX,.CRT);
188   4               Wait for (.CRT$EXCH,0);
189   4           end;
          /* test for request to exit this mode */
190   3           MSG$PTR=RQACPT (.READ$BUFFER$EXCH);
191   3           If ALARM=0
                  then do;
193   4               If (MSG$PTR <> 0 and BUFFERA(0) = 1BH)
                      then do;
195   5                   MSG$PTR=RQWAIT(.CRT$DISPLAY$LOCK,0);
196   5                   start task (.UPDATE$EXCH,MSG$PTR);
197   5               end;
198   4               else do;
199   5                   If MSG$PTR=0
                          then STARTER(2).TYPE=200;
201   5                   else STARTER(2).TYPE=100;
202   5                   Start task (.CRT$STATUS$EXCH,.STARTER(2));
203   5                   NEW=0;
204   5               end;
205   4           end;
206   3       end;
207   2   end CRT$DATA$TASK;
208   1 end CRT$DATA$MODULE;


MODULE INFORMATION:
     CODE AREA SIZE     = 0720H    1824D
     VARIABLE AREA SIZE = 0189H     393D
     MAXIMUM STACK SIZE = 0004H       4D
     388 LINES READ
     0 PROGRAM ERROR(S)

END OF PL/M-80 COMPILATION
```

```
        $TITLE('ASCII STRING TO FIXED BINARY')
        /*******************************************************
        * This program converts an ASCII string into a fix- *
        * ed point binary number.  The fixed decimal point  *
        * is determined by the parameter passed in SIZE.    *
        *******************************************************/
1       ASC$2$BINARY$MODULE:
        Do;
        /* SPECIAL ASCII CHARACTERS */
2   1   DECLARE
          NULL          LITERALLY '00H',
          CONTROL$C     LITERALLY '03H',
          CONTROL$E     LITERALLY '05H',
          BELL          LITERALLY '07H',
          TAB           LITERALLY '09H',
          LF            LITERALLY '0AH',
          VT            LITERALLY '0BH',
          FF            LITERALLY '0CH',
          CR            LITERALLY '0DH',
          CONTROL$P     LITERALLY '10H',
          CONTROL$Q     LITERALLY '11H',
          CONTROL$R     LITERALLY '12H',
          CONTROL$S     LITERALLY '13H',
          CONTROL$X     LITERALLY '18H',
          CONTROL$Z     LITERALLY '1AH',
          ESC           LITERALLY '1BH',
          QUOTE         LITERALLY '22H',
          LCA           LITERALLY '61H',
          LCZ           LITERALLY '7AH',
          RUBOUT        LITERALLY '7FH';

3   1   ASC$2$BINARY:
        Procedure (SRC$PTR,TRGT$PTR,SIZE) byte public;
4   2       Declare (SRC$PTR,TRGT$PTR) address;
5   2       Declare (SOURCE based SRC$PTR)(80) byte;
6   2       Declare RESULT based TRGT$PTR address;
7   2       Declare (N,SIZE,K,DP,DIGITS,VALID) byte;
8   2       Declare POWER(6) address data (
                0,1,10,100,1000,10000 );
          /* Find location of decimal point */
9   2         n=0;
10  2         Do while SOURCE(n)<>'.' AND SOURCE(n)<>CR
                    AND SOURCE(n)<>LF;
11  3             n=n+1;
12  3           end;
13  2         DP=n;
          /* Provide correct number of digits to right of decimal */
14  2         Do n=0 to SIZE;
15  3             SOURCE(DP+n)=SOURCE(DP+n+1);
16  3             If SOURCE(DP+n)>39H OR SOURCE(DP+n)<30H
                  then do k=n to SIZE;
18  4                 SOURCE(DP+k)='0';
19  4             end;
```

```
20    3           end;
                /* Mark end of string */
21    2           DIGITS=DP+SIZE;
                /* Test for all valid characters */
22    2           VALID=1;
23    2           Do n=0 to DIGITS;
24    3               If SOURCE(n)>39H OR SOURCE(n)<30H
                        then VALID=0;
26    3           end;
27    2           If DIGITS>5
                    then VALID=0;
                /* Convert data to binary and store */
29    2           n=0;
30    2           If VALID=1
                    then do;
32    3               RESULT=0;
33    3               Do while DIGITS > 0;
34    4                   RESULT=RESULT+((
                                SOURCE(n) AND 0FH) * POWER(DIGITS));
35    4                   n=n+1;
36    4                   DIGITS=DIGITS-1;
37    4               end;
38    3           end;
                /* Return to calling program */
39    2           Return VALID;
40    2         end ASC$2$BINARY;
41    1       end ASC$2$BINARY$MODULE;


MODULE INFORMATION:

     CODE AREA SIZE      = 0178H     376D
     VARIABLE AREA SIZE  = 00CAH     10CD
     MAXIMUM STACK SIZE  = 0004H       4D
     80 LINES READ
     0 PROGRAM ERROR(S)

END OF PL/M-80 COMPILATION
```

```
                $TITLE('COMMON VARIABLE STORAGE')
                /*************************************************
                * This module contains those variables common to  *
                * multiple tasks in the oven control application. *
                *************************************************/
  1             VARIABLE$STORAGE:
                Do;
  2     1       Declare SETPOINT(4) address public;
  3     1       Declare TOLERANCE(4) address public;
  4     1       Declare TEMP(4) address public;
  5     1       Declare STATUS(4) byte public;
  6     1       Declare BLOCK0 byte public;
  7     1       Declare BLOCK1 byte public;
  8     1       Declare BLOCK2 byte public;
  9     1       Declare BLOCK3 byte public;
 10     1       end VARIABLE$STORAGE;


MODULE INFORMATION:

     CODE AREA SIZE     = 0000H      0D
     VARIABLE AREA SIZE = 0020H     32D
     MAXIMUM STACK SIZE = 0000H      0D
     16 LINES READ
     0 PROGRAM ERROR(S)

END OF PL/M-80 COMPILATION
```

```
                $TITLE('WORD TO ASCII CONVERSION')
                /*******************************************************
                * This routine converts a fixed point word in mem-     *
                * ory into a 4 digit plus 1 decimal ASCII display-     *
                * able number.  Zero blanking is included.             *
                *******************************************************/
  1             DEC$REP$MODULE:
                Do;

  2   1         DEC$REP:
                Procedure (SOURCE,TARGET) public ;
  3   2           Declare (SOURCE,TARGET) address;
  4   2           Declare ANSWR(5) byte;
  5   2           Declare (DISPLAY based TARGET)(5) byte;
  6   2           Declare NUMBER based SOURCE structure (
                      ELEMENT address );
  7   2           Declare N byte;
  8   2           Declare CALC(5) address;
                /* Initialize */
  9   2           Do n=0 to 4;
 10   3               ANSWR(n)='0';
 11   3           end;
 12   2           CALC(0)=NUMBER.ELEMENT;
                /* Convert to ASCII */
 13   2           Do n=1 to 5;
 14   3               CALC(n)=CALC(n-1)/10;
 15   3               ANSWR(5-n)=(CALC(n-1) mod 10) + 30H;
 16   3           end;
                /* Perform zero blanking */
 17   2           Do n=0 to 3;
 18   3               If ANSWR(n)<>'0'
                     then n=4;
 20   3               else ANSWR(n)=' ';
 21   3           end;
                /* Format with decimal point */
 22   2           Call move (4,.ANSWR,TARGET);
 23   2           DISPLAY(4)='.';
 24   2           DISPLAY(5)=ANSWR(4);
 25   2         end DEC$REP;
 26   1       end DEC$REP$MODULE;


MODULE INFORMATION:

     CODE AREA SIZE     = 00EEH     238D
     VARIABLE AREA SIZE = 0014H      20D
     MAXIMUM STACK SIZE = 0004H       4D
     40 LINES READ
     0 PROGRAM ERROR(S)
END OF PL/M-80 COMPILATION
```

Distributed Control Modules 14

INTRODUCTION TO THE DISTRIBUTED CONTROL MODULES

Overview

Intel's Distributed Control Module (iDCM) products provide building blocks for construction of real-time distributed control systems based on the BITBUS™ interconnect. This new serial bus architecture addresses many of the limitations inherent in traditional connection methods. For instance, future system cost reductions are limited in systems based on parallel bus structures due to their electrical and mechanical characteristics. Other traditional connection methods such as current loops and RS 232 C do not provide sufficient performance or flexibility for complex industrial control applications. In addition, although there are numerous industry standards for connecting microprocessors, the MULTIBUS® and the STD-bus for example, there is no standard connection for microcontrollers. The BITBUS interconnect (Table 1) combines existing standards with new standard interfaces to provide the optimal solution for difficult distributed control problems.

The iDCM products combine hardware and software for use in applications that would benefit most from employing distributed architectures. Applications such as robotics, process control, data acquisition and control, and environmental control are a few examples.

Table 1. Standard BITBUS™ Interfaces

Interface	Specification
Electrical	RS485
Cable	10-conductor flat ribbon or 1 to 2 wire twisted pair
Back-plane connector	64-pin Standard DIN
End-cable connector	3M #3446-1302 female
Control-board form-factor	Single-height, Double-depth Eurocard
Data Link control	Synchronous Data-link Control
Data transfer rate	62.5K baud, 375K baud and 2.4M baud
Message formats	Compatible with iRMX 51 format command/response/status
Common command sequences	Integral Remote Access and Control (RAC) function
Operating systems	S/W drivers for iRMX 86, 88, 286R and ISIS (for iPDS only)

Benefits of Distributed Architectures

Distributed architectures are intrinsically more reliable than centralized architectures. In a centralized control system a central controller failure results in a system-wide failure. Distributed systems can be configured to prevent this. Also, distributed systems are more cost effective and more easily modified. For instance, performance improvements in centralized systems are expensive and do not concentrate improvements in the areas where they are needed most. In distributed systems, only the specific parts of the system that require enhancement need be modified. Most importantly, control systems based on distributed architectures have less difficulty responding to the external environment because they have less to manage.

The BITBUS™ Interconnect

The BITBUS interconnect is a serial bus optimized for high speed transfer of short control messages in a hierarchical system. In order to provide an easy to use high performance serial interconnect, transparent to the applications programmer, high-level interfaces are specified. These interfaces include: the message structure and protocol for a multitasking environment, and a set of high-level commands for remote I/O access and application task control. As with traditional bus specifications, the electrical and data protocol levels have been defined.

The BITBUS interconnect supports up to 250 nodes and three bit rates dependent on application performance requirements. Different BITBUS segments may support different bit rates.

A Simple and Reliable Solution

The BITBUS architecture supplies the system designer with a simple and reliable foundation. Some key features of this architecture are: defined high-level interfaces that provide all communication and user program management, the reliable SDLC protocol, power-up diagnostics, standard industrial packaging, compact software and hardware provided in the high performance 12MHz 8044 microcontroller, and a board-level integrated solution. In addition, complex, expensive, and awkward connection problems are no longer a factor because the BITBUS interconnect is a serial bus requiring a simple twisted wire pair.

Open Systems — An Answer to Obsolescence

Intel's Open Systems philosophy requires systems be open to: future VLSI, all levels of integration, third party suppliers, and special requirements. In order to facilitate this design strategy, the BITBUS interconnect was developed as a standard microcontroller interface. The same benefits realized by users of Intel's MULTIBUS architecture will be realized by users of the BITBUS architecture — the ability to exploit VLSI technology without having to pay premiums for new system design, multiple supply sources, wide product selections, and competitive prices.

The Open Systems philosophy characterizes the iDCM product line. Distributed Control Modules are compatible (open) at three levels of integration: components, boards, and systems. This multilevel approach enables OEM's to adapt to new business environments and opportunities as VLSI technology evolves.

Distributed Control Modules

The iDCM product line consists of both software and hardware products: the iRMX™ 51 Executive, iRMX 510 Support Package, iSBX™ 344 BITBUS MULTIMODULE™ Board, and the iRCB 44/10 BITBUS Remote Controller Board. All iDCM hardware products include integral firmware to implement the high-level BITBUS interfaces: message formats, command sequences, and operating system environments.

iRMX™ 51 Executive

The iRMX 51 Executive is a compact, easy to use, software tool for development and implementation of applications built on the high performance 8-bit family of 8051 microcontrollers. A pre-configured version of the Executive is included in firmware of the two iDCM hardware products. During run-time, some of the services provided by this event driven Executive are: task scheduling, interrupt handling, and message passing. Streamlined code, the simple user interface and modular design of the iRMX 51 Executive enhance system reliability.

iRMX™ 510 iDCM Support Package

The iRMX 510 iDCM Support Package provides the software development and run-time support for BITBUS systems. Also included are the software interfaces for other operating systems: iRMX 86, iRMX 286R, iRMX 88, and the iPDS ISIS. These software interfaces ease integration of a BITBUS system into MULTIBUS or iPDS environments.

iSBX™ 344 BITBUS™ MULTIMODULE™ Board

The iSBX 344 board facilitates expansion of MULTIBUS and iPDS systems via the BITBUS interconnect. This board is the iDCM hardware interface. MULTIBUS system capabilities can be expanded to include low-cost remote control using the iSBX 344 MULTIMODULE board and the iRMX 510 iDCM Support Package. Also, BITBUS system capabilities can be expanded using this board with user supplied software. The iSBX 344 board's integral firmware reduces application development time, ensures real time response, eases system integration, and lowers system cost.

iRCB 44/10 BITBUS™ Remote Controller Board

The iRCB 44/10 BITBUS Remote Controller module is a low-end, single-board computer with 24 lines of parallel I/O. The board has a single-high Eurocard form factor with a DIN connector for increased reliablity and integration with standard industrial packaging. One iSBX I/O Expansion connector will accommodate one of many iSBX MULTIMODULE Boards for I/O expansion. Also, sockets for repeaters are provided for extending the BITBUS interconnect beyond the length limits of one BITBUS segment. This board lowers distributed system cost via the BITBUS interconnect support and the same integral firmware provided on the iSBX 344 BITBUS Controller MULTIMODULE board.

Expanding a MULTIBUS® System with Distributed Control Modules

An example of how a MULTIBUS system can be expanded with iDCM Modules is shown in Figures 1 and 2. Figure 1 shows a basic MULTIBUS system: processor board, memory module, and I/O controller. Figure 2 illustrates the expanded system. Some advantages of the expanded system follow: The burden on the central processor has been reduced, thereby increasing overall system performance. System cost reduction is realized because the BITBUS architecture removes the necessity of adding expensive centralized systems to handle increased performance demands. Also, the BITBUS architecture enables implementation of a more efficient and flexible system that is insensitive to the addition of more nodes, or changes in node job functions.

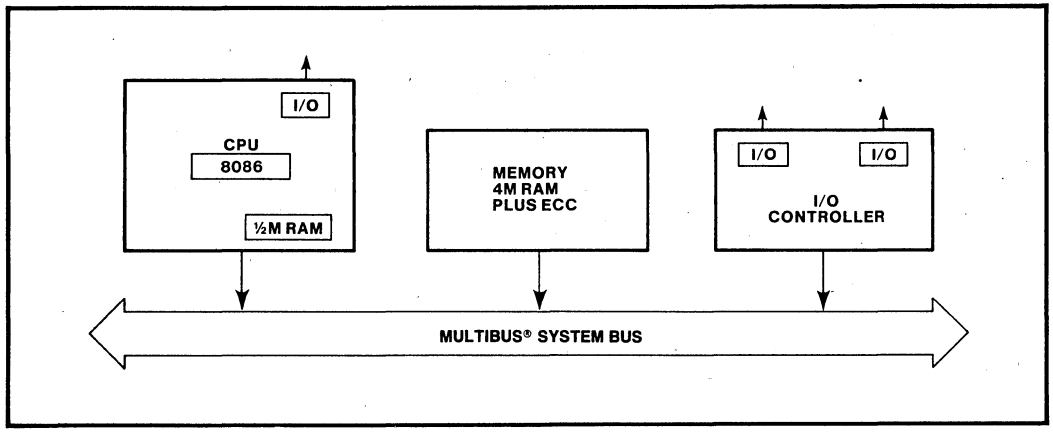


Figure 1. Basic MULTIBUS® System

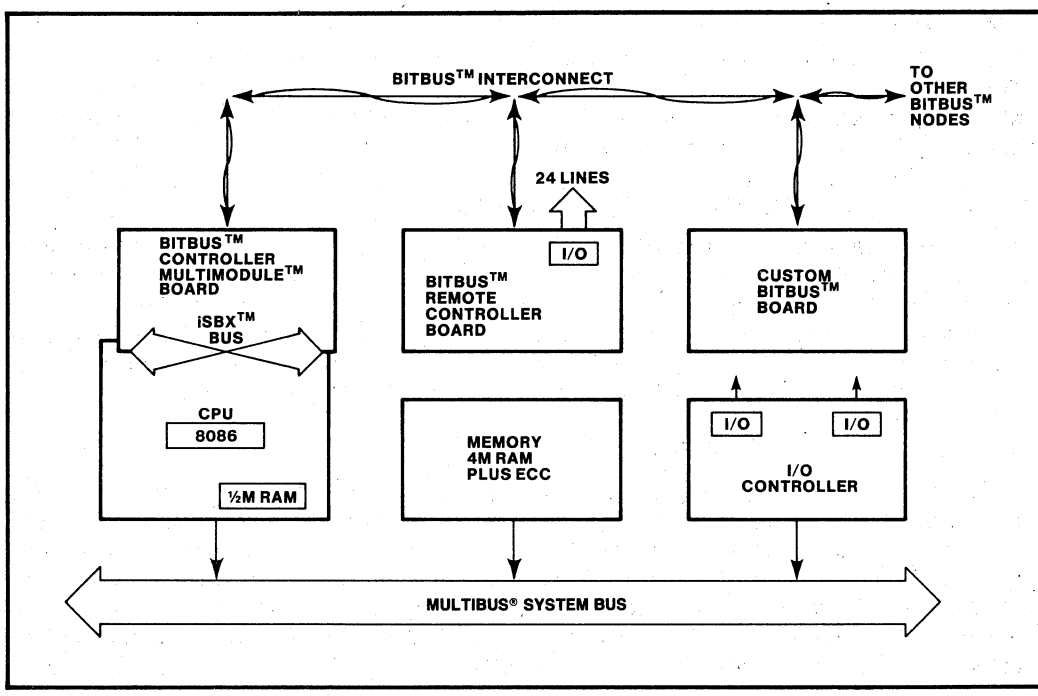


Figure 2. MULTIBUS® System Expanded with BITBUS™ Interconnect and Distributed Control Modules

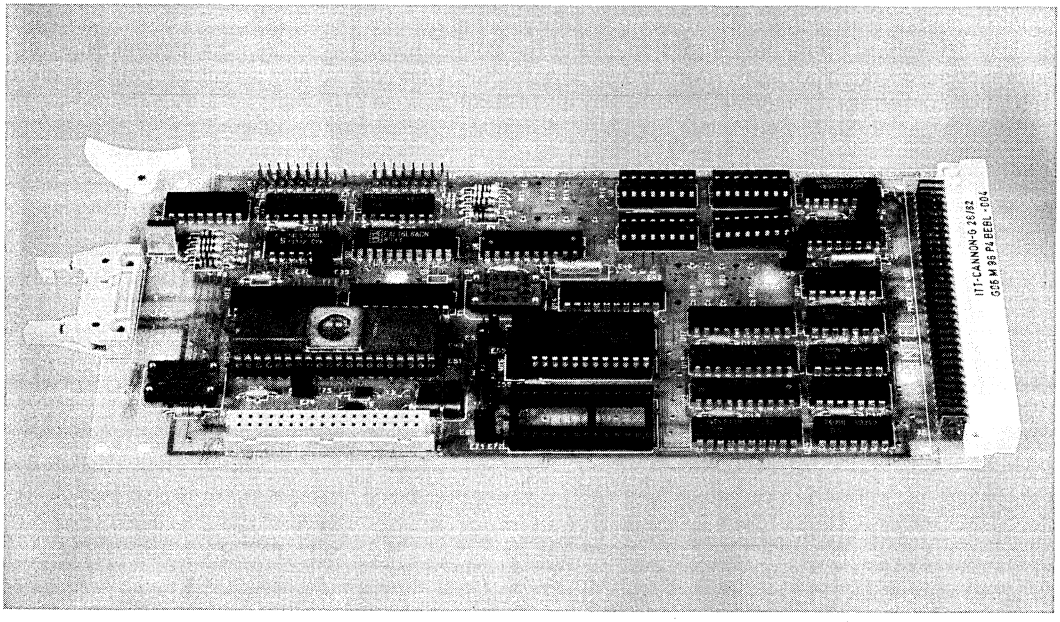
Summary

This overview has introduced the attributes and advantages of the BITBUS interconnect and Distributed Control Modules. Initial iDCM products (the iRMX 51 Executive, the iRMX 510 iDCM Support Package, iSBX 344 BITBUS Controller MULTIMODULE and iRCB 44/10 BITBUS Remote Controller boards) are intended to allow rapid assimilation of this new technology. Data sheets describing the individual iDCM products are included in the next section of this document. The final section presents the BITBUS Specification (supported by the iDCM products).

iRCB 44/10
BITBUS™ Remote Controller Board

- **High performance 12 MHz 8044 controller**
- **Integral firmware including the iRMX™ 51 Executive optomized for real-time control**
- **Full BITBUS™ support**
- **Standard industrial packaging: Eurocard, DIN connector**
- **2 28-pin JEDEC memory sites for user's control functions**
- **Low cost I/O expansion with 8-bit iSBX™ connector**
- **Programmable control/monitoring capability of 24 I/O lines**
- **Power up diagnostics increase reliability**

The iRCB 44/10 BITBUS™ Remote Controller Board, is an intelligent real-time controller and a remote I/O expansion device. Based on the highly integrated 8044 component (an 8 bit 8051 microcontroller and an intelligent SDLC controller on one chip) the iRCB 44/10 board provides high performance control capability at low cost. Incorporating complete BITBUS support, the iRCB 44/10 board and the other members of Intel's Distributed Control Modules (iDCM) family expand Intel's OEM microcomputer system capabilities to include distributed real-time control. Like all members of the iDCM family, the iRCB 44/10 board includes many features that make it well suited for industrial control applications such as: data acquisition and monitoring, process control, robotics, and machine control.



OPERATING ENVIRONMENT

Intel's Distributed Control Modules (iDCM) product family contains the building blocks to implement real-time distributed control applications. The iDCM family incorporates the BITBUS interconnect to provide standard high speed serial communication between microcontrollers. The two iDCM hardware products, the iSBX 344 BITBUS controller MULTIMODULE™ board and the iRCB 44/10 BITBUS Remote Controller Board, communicate in an iDCM system via the BITBUS interconnect as shown in Figure 1.

The iRCB 44/10 board can be used as an intelligent remote controller or an I/O expansion device. When performing as an intelligent controller the iRCB 44/10 board not only monitors the status of multiple process points, but it can execute varied user supplied control algorithms. When functioning as an I/O expansion device, the iRCB 44/10 board simply collects data from multiple I/O ports and transmits this information via the BITBUS or iSBX bus interface to the system controller for analysis or updating purposes.

As a member of the iDCM product line the iRCB 44/10 board fully supports the BITBUS microcontroller interconnect. Typically, the iRCB 44/10 board would be a node in a BITBUS system. The iRCB 44/10 board could be part of a master or slave node. (The BITBUS system supports a multidrop configuration: one master, many slaves.)

ARCHITECTURE

Figure 2 illustrates the major functional blocks of the iRCB 44/10 board: iDCM controller, memory, BITBUS microcontroller interconnect, parallel I/O, iSBX expansion, initialization and diagnostic logic.

iDCM Controller

The heart of the iRCB 44/10 board's controlling and communication capability is the highly integrated 12 MHz 8044 microcontroller. The 8044 consists of the advanced 8-bit 8051 microcontroller and a SDLC controller called the Serial Interface Unit (SIU). This dual processor architecture allows complex control and high speed communication functions to be realized cost effectively.

Another essential part of the iDCM controller is the integral firmware that resides on-chip to implement the BITBUS interface. In the operating environment of the iRCB 44/10 board, the 8044's SIU acts as a SDLC controller which offloads the on-chip 8051 microcontroller of communication tasks; freeing the 8051 to concentrate on real-time control.

The iDCM controller (8044 microcontroller and on-chip firmware) provides, in one package, a simple user interface, and high performance communications and control capabilities to efficiently and economically build a complex control system.

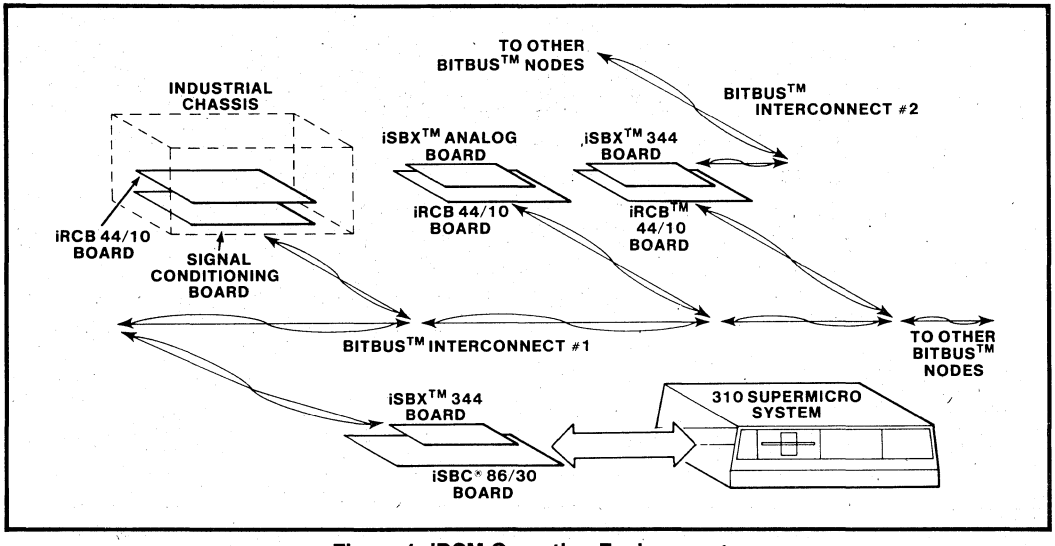


Figure 1. iDCM Operating Environment

Memory

The iRCB 44/10 board memory consists of two sections: internal and external. Internal memory is located in the on-chip memory of the iDCM controller. The iRMX 51 Executive and the remaining iDCM controller firmware ration this resource. However, eight bytes of bit addressable internal memory are reserved for the user. Ample space is reserved for user programs and data in the iRCB 44/10 board external memory.

Two 28 pin JEDEC sites comprise the iRCB 44/10 board external memory. One site has been dedicated for data, the other for code. Table 1 lists the supported memory devices for each site. Intel's 2764, 27128, and 2817A are a few examples. The user may choose one of two memory configurations and specify different memory sizes by placing the proper jumpers at system initialization. The most flexible configuration option provides the user with access to the code site for program download or upload. This feature ensures expansion of an existing system is easily accomodated.

Table 1. Supported Memory Devices

DEVICE	DATA SITE	CODE SITE
4K x 8-64K x 8 EPROM/ROM	NO	YES
2K x 8-32K x 8 SRAM	YES	YES
2K x 8-16K x 8 NVRAM and E2PROM	NO	YES

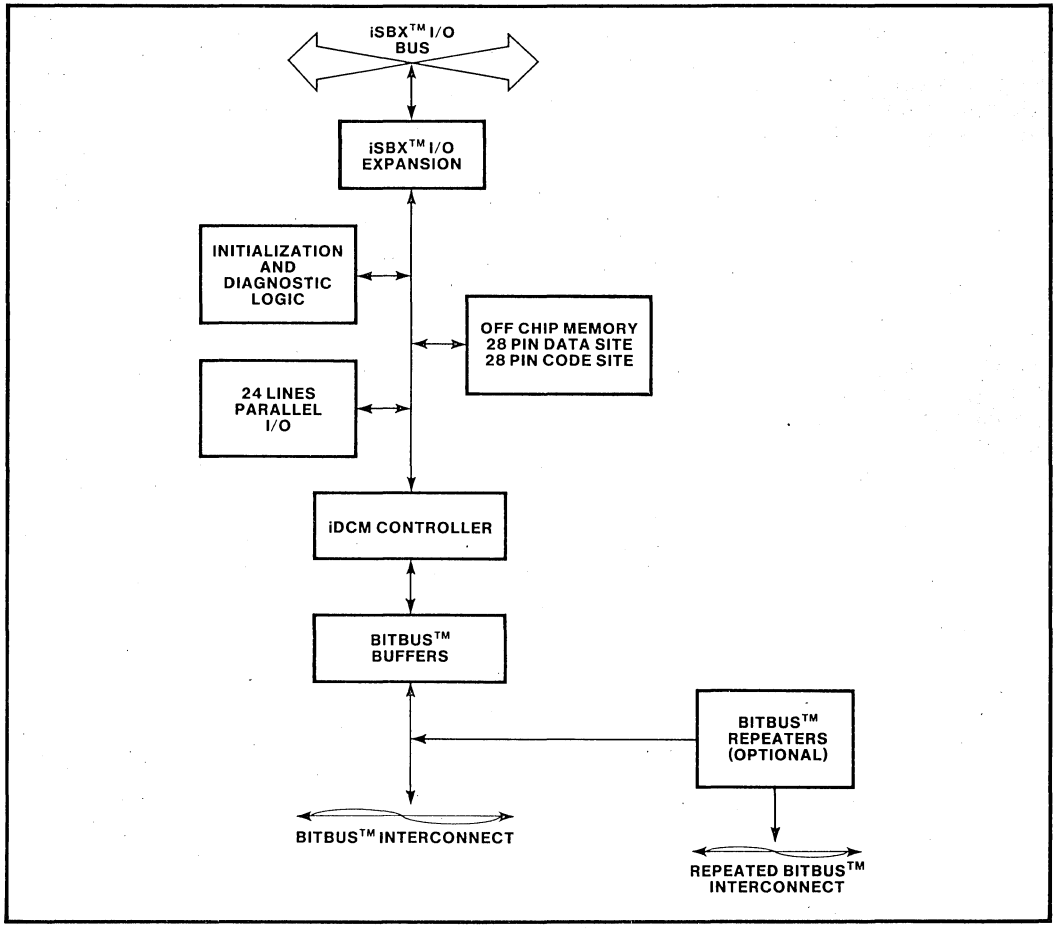


Figure 2. iRCB™ 44/10 Block Diagram

BITBUS™ Microcontroller Interconnect

The iRCB 44/10 board serial interface fully supports the BITBUS microcontroller interconnect. The BITBUS interconnect is a serial bus optimized for control applications. The bus supports both synchronous and self-clocked modes of operation. These modes of operation are selectable dependent on application requirements as are the transmission speeds. Table 2 shows the different combinations of modes of operation, transmission speeds, and distances. The SDLC protocol, BITBUS message format, and compatibility with Intel's other software and hardware products comprise the remainder of the BITBUS architecture. These features contribute to BITBUS system reliability and usefulness as a microcontroller interconnect.

The BITBUS connection consists of one or two differential pair(s) of wires. The serial (BITBUS) interface of the iRCB 44/10 board consists of : a half-duplex RS 485 tranceiver, an optional BITBUS repeater and an optional clock source for the synchronous mode of operation.

Parallel I/O

In order to provide an optimal parallel I/O interface for control applications, the iRCB 44/10 board supports 24 software programmable parallel I/O lines. This feature supplies the flexibility and simplicity required for control and data acquisition systems. Sixteen of these lines are fully programmable as inputs or outputs, with loopback, on a bit by bit basis so that bit set, reset, and toggle operations are streamlined. The remaining eight lines are dedicated as inputs. Figure 3 depicts the general I/O port structure.

The parallel I/O lines can be manipulated by using the Remote Access and Control (RAC) function (in iDCM controller firmware) from a supervisory node or locally by a user program. The user program can also access the RAC function or directly operate the I/O lines. Input, output, mixed — input and output, and bit operations are possible simply by reading or writing a particular port.

iSBX™ Expansion

One iSBX I/O expansion connector is provided on the iRCB 44/10 board. This connector can be used to extend the I/O capability of the board. In addition to specialized and custom designed iSBX boards, a full line of compatible high speed, 8-bit expansion MULTIMODULE boards, both single and double wide, are available from Intel. The only incompatible modules are those that require the MWAIT* signal or DMA operation. A few of Intel's iRCB 44/10 board compatible iSBX MULTIMODULE boards include: parallel I/O, serial I/O, BITBUS expansion, IEEE 488 GPIB, analog input, analog output, and magnetic bubble.

With the iSBX 344 BITBUS Controller MULTIMODULE board and user supplied software, the iRCB 44/10 board can act as an intelligent BITBUS repeater facilitating the transition between two BITBUS segments operating at different speeds.

Initalization and Diagnostic Logic

Like the other members of Intel's Distributed Control Modules (iDCM) product line, the iRCB 44/10 board includes many features which make it well suited for industrial control applications. Power up diagnostics is just one of these features. Diagnostics simplify system startup considerably, by immediately indicating an iDCM controller or external bus failure. The LEDs used for power up diagnostics are available for user diagnostics after power up as well to further contribute to reliable operation of the system.

Initial iRCB 44/10 board parameters are set by positioning jumpers. The jumpers determine the

Table 2. Modes of Operation

	Speed Kb/s	Maximum Distance Between Repeaters M/ft	Maximum # Nodes Between Repeaters	Maximum # Repeaters
Synchronous	2400	30/100	28	0
Self Clocked	375	300/1000	28	2
	62.5	1200/4000	28	10

BITBUS mode of operation: synchronous, self clocked, transmission speed, and address of the iRCB 44/10 board in the BITBUS system. This minimizes the number of spare boards to be stocked for multiple nodes, decreasing stocking inventory and cost.

INTEGRAL FIRMWARE

The iRCB 44/10 board contains resident firmware located in the iDCM Controller. The on-chip firmware consists of: a pre-configured iRMX 51 Executive for user program development; a Remote Access and Control (RAC) function that enables user communication and control of different microcontrollers and I/O points; a communications gateway to connect the BITBUS interconnect, iSBX bus, and iRMX 51 tasks; and power up diagnostics.

The iRMX 51 Executive is an event-driven software manager that can respond to the needs of multiple tasks. This real-time multitasking executive provides: task management, timing, interrupt handling, and message passing services. Table 3 shows the iRMX 51 user interfaces. Both the Executive and the communications gateway allow for the addition of up to seven user tasks at each node while making BITBUS operations transparent.

The Remote Access and Control Function is a special purpose task that allows the user to trans-

Table 3. iRMX™ 51 Interfaces

COMMAND	DESCRIPTION
RQ SEND MESSAGE	Sends a message (a command from the BITBUS master, a response from a slave, or a simple message between tasks on the same BITBUS component) to another task.
RQ WAIT	Waits for an interrupt, and event time-out, a message, or any combination of the three.
RQ CREATE TASK	Causes a new sequence of code to be run as an iRMX 51 task with a specific function identification code and priority.
RQ DELETE TASK	Stops the specified task and removes it from all execution lists.
RQ ALLOCATE	Allocates a fixed-length buffer from the on-chip, scratch-pad RAM for general use, or, in BITBUS applications, for a BITBUS message buffer.
RQ DEALLOCATE	Returns an on-chip buffer to the system.
RQ SET INTERVAL	Set the time interval to be used as a separate event-timer for the task.
RQ ENABLE INTERRUPT	Allow external interrupts to signal the microcontroller.
RQ DISSABLE INTERRUPT	Stops all external interrupts from signalling the microcontroller.
RQ GET FUNCTION ID	Provides a list of the 8 function identification codes representing the tasks currently operating on the microcontroller.

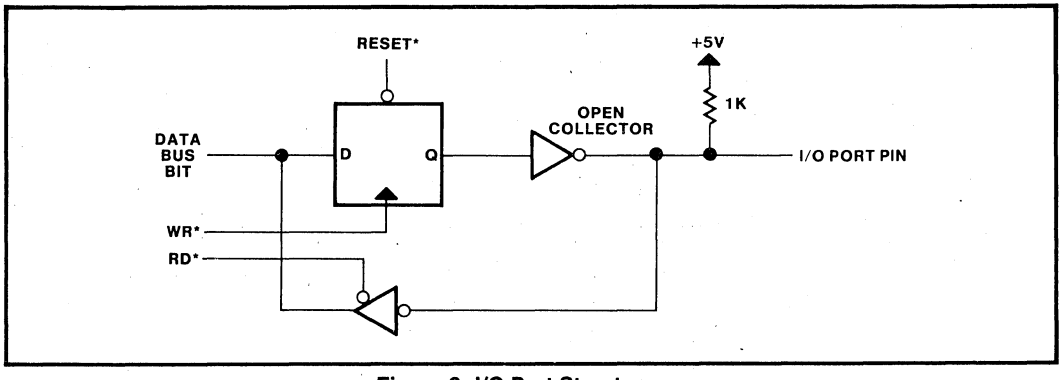


Figure 3. I/O Port Structure

fer commands and program variables to remote BITBUS controllers, obtain the status of a remote I/O line(s), or reverse the state of a remote I/O line. Table 4 provides a complete listing of the RAC services. No user code need be written to use this function. Power up tests provide a quick diagnostic service.

The services provided by the iRCB 44/10 board integral firmware simplify the development and implementation of complex real-time control application systems. All iDCM hardware products contain integral firmware thus supplying the user with a total system solution.

INDUSTRIAL PACKAGING

The iRCB 44/10 form factor is a single high, 220mm deep Eurocard as shown in Figure 4. The Eurocard form factor supports most standard industrial packaging schemes as well as Intel's MULTIBUS® II packaging scheme. The Eurocard form factor specifies reliable DIN connectors. A standard 64 pin connector is included on the iRCB 44/10 board.

DEVELOPMENT ENVIRONMENT

Intel provides a complete development environment for the iRCB 44/10 board. Software development support consists of: the 8051 Software Development Package, and the iRMX 510 iDCM Support Package. The 8051 Software Development Package provides the RL 51 Linker and Relocator Program, and ASM 51. PL/M 51 is also available. The iRMX 510 Support Package includes the iDCM Controller firmware files on diskette, as well as iRMX 51 libraries. Hardware tools consist of the In-Circuit Emulator (ICE-44), Intel's Portable Development System (iPDS with EMV-44), and Intellec Series II or III Development Systems.

Table 4. RAC Services

COMMAND	DESCRIPTION
READ I/O	Read external I/O location. Return result in reply message.
WRITE I/O	Write byte to external I/O location.
UPDATE I/O	Write byte to, then read byte from external I/O location. Return result in reply message.
OR I/O	OR data with contents of external I/O location. Return OR'd value.
AND I/O	AND data with contents of external I/O location. Return AND'd value.
XOR I/O	XOR data with contents of external I/O location. Return XOR'd value.
READ INTERNAL MEMORY	Read contents of internal memory location. Return result in reply message.
WRITE INTERNAL MEMORY	Write data to internal memory location.
DOWNLOAD EXTERNAL MEMORY	Write data starting at external memory location.
UPLOAD EXTERNAL MEMORY	Read data starting at external memory location. Return result in reply message.
GET FUNCTIONS	Provides a list of the 8 function identification codes representing the tasks currently operating on the microcontroller.
CREATE TASK	Causes a new sequence of code to be run as in the iRMX 51 interface.
DELETE TASK	Stops the specified task and removes it from all execution lists as in the iRMX 51 interface.
RAC PROTECT	Suspends or resumes RAC Services.
RESET DEVICE	Returns device software to original state at initialization.

NOTES:
Internal memory locations are included in the 192 bytes of data RAM provided in the microcontroller. External memory refers memory outside the microcontroller — the 28-pin sockets of the iSBX 344 module and the iRCB 44/10 board. Each RAC Access Function may refer to 1, 2, 3, 4, 5, or 6 individual I/O or memory locations in a single command.

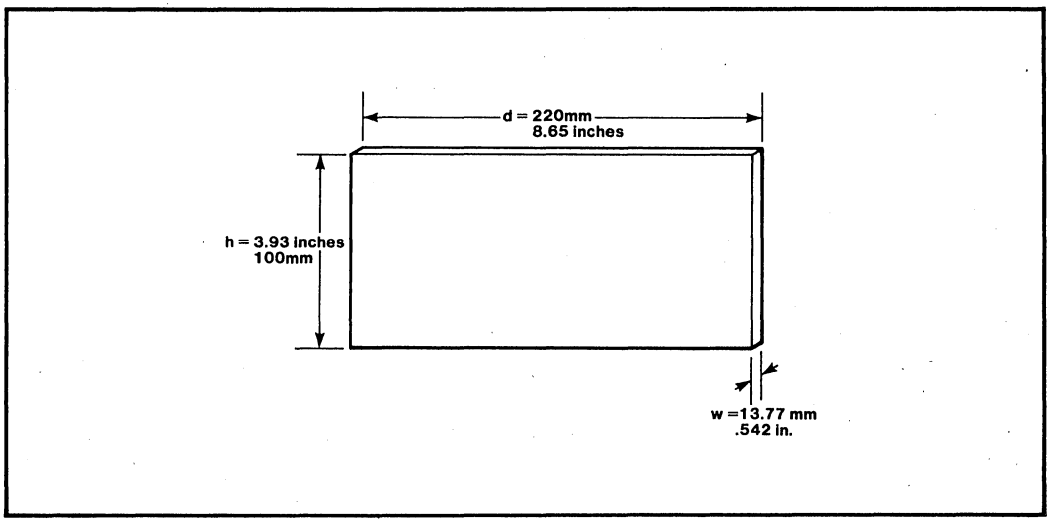


Figure 4. Eurocard Single High Form Factor

SPECIFICATIONS

Word Size

Instruction — 8 bit
Data — 8 bit

Processor Clock 12 MHz

Instruction Execution Times
1 μsec 60% instructions
2 μsec 40% instructions
4 μsec Multiply & Divide

Memory Capacity/Addressing

iDCM Controller — 64 K

Address Range

	Option 1	Option 2
External Memory		
Data	0000H-7FFFH	0000H-7FFFH
Code	1000H-0FFFFH	8000H-0FFEFH
Internal Memory		
Code	0000H-0FFFH	0000H-0FFFH

I/O Capacity

iSBX MULTIMODULE™ board — one single or doublewide not requiring MWAIT* or DMA
24 Digital Lines Programmable Parallel I/O

Interrupt Sources

Two external — iSBX I/O Expansion bus sources or other sources.
BITBUS Microcontroller Interconnect.

Terminations

Sockets provided on board for ¼ Watt 5% Carbon type resistors. Resistor value to match characteristic impedance of cable as closely as possible — 120 ohms or greater.

Repeaters

Sockets provided on board — Devices 75174 and 75175

Connector Options

10 Pin Plug

Flat Cable — 3M 3473-6010, TB Ansley 609-1001M, or equal
Discrete Wire — BERG 65846-007, ITT Cannon 121-7326-105, or equal

DIN Connector Plug

Flat Cable — GW Elco 00-8259-096-84-124, Robinson Nugent RNE-IDC64C-TG30, or equal
Discrete Wire — ITT Cannon G06 M96 P3 BDBL-004 GW Elco 60 8257 3017, or equal

iDCM Controller I/O Addressing

FUNCTION	ADDRESS	READ	WRITE	BIT
PORT A	FFCOH	✓	✓	
PORT B	FFC1H	✓		
PORT C	FFC2H	✓	✓	
MCSO	FF80H-FF87H FF00, FF01	✓	✓	
MSC1	FF88H-FF8F	✓	✓	
LED #1	90H	✓	✓	✓
LED #2	91H	✓	✓	✓
RDY/NE*	B4H	✓	✓	✓
NODE ADDRESS	FFFFH	✓		
CONFIGURATION	FFFEH	✓		
OPT0	92H	✓	✓	✓
OPT1	93H	✓	✓	✓
INT0	B2H	✓		✓
INT1	B3H	✓		✓

10 Pin Repeater Connector Pin Out

PIN	SIGNAL
1	+12V
2	+12V
3	GND
4	GND
5	DATA*
6	DATA
7	DCLK*/RTS*
8	DCLK/RTS
9	RGND
10	RGND

iRCB 44/10 Pin Out

DIN Connector

DIN PIN #	PIN & SOCKET PIN #	FUNCTION	DIN PIN #	PIN & SOCKET PIN #	FUNCTION
1a		GND	1c		GND
2a		+5V	2c		+5V
3a		DATA	3c		DATA*
4a		DLCK/RT	4c		DLCK*/RTS*
5a	1	EXTINT	5c		RGND
6a	3	PB7	6c	2	GND
7a	5	PB6	7c	4	GND
8a	7	PB5	8c	6	GND
9a	9	PB4	9c	8	GND
10a	11	PB3	10c	10	GND
11a	13	PB2	11c	12	GND
12a	15	PB1	12c	14	GND
13a	17	PB0	13c	16	GND
14a	19	PC3	14c	18	GND
15a	21	PC2	15c	20	GND
16a	23	PC1	16c	22	GND
17a	25	PC0	17c	24	GND
18a	27	PC4	18c	26	GND
19a	29	PC5	19c	28	GND
20a	31	PC6	20c	30	GND
21a	33	PC7	21c	32	GND
22a	35	PA7	22c	34	GND
23a	37	PA6	23c	36	GND
24a	39	PA5	24c	38	GND
25a	41	PA4	25c	40	GND
26a	43	PA3	26c	42	GND
27a	45	PA2	27c	44	GND
28a	47	PA1	28c	46	GND
29a	49	PA0	29c	48	GND
30a		+12V	30c		−12V
31a		+5V	31c		+5V
32a		GND	32c		GND

Electrical Characteristics

Interfaces

iSBX I/O expansion bus — supports the standard I/O Expansion Bus Specification with compliance level D8/8F

Memory Sites — Both code and data sites support the electrical Universal Memory Site specification

BITBUS™ Interconnect — The iRCB 44/10 Remote Controller Board supports the BITBUS Specification as follows:
Fully supported synchronous mode at 2.4 Mbits/second and self clocked mode for 375 kbits/second and 62.5 kbits/second.

The iRCB 44/10 Remote Controller Board presents one standard load to the BITBUS bus without repeaters, with repeaters two standard loads

Message length of 18 bytes supported

RAC Function support as shown in Table 4

Parallel I/O — See the Table 5 for Electrical Specifications of the interface.

Power Requirements

.9A at +5V ± 5% iRCB 44/10 board only — memory, repeater, or iSBX board NOT included

Physical Characteristics

Single high, 220mm deep Eurocard Form Factor

Dimensions

Width — 13.77 mm (.542 in) maximum component height

Height — 100 mm (3.93 in)

Depth — 220 mm (8.65 in)

Weight — 169 gm (6 ounces)

Environmental Characteristics

Operating Temperature — 0°C to 55°C at 200 Linear Feet/Minute Air Velocity

Humidity — 90% non-condensing

Reference Manual (NOT Supplied)

146312 — Guide to Using the Distributed Control Modules

Table 5. Parallel I/O Electrical Specification

PARAMETER	CONDITION	MIN	MAX	UNITS
V_{OL}	$I_{OL} = 16$ mA		0.5	V
V_{OH}	$I_{OH} = -2$ mA	2.4		V
V_{IH}		2.0	7.0	V
V_{IL}		−1.0	0.8	V
I_{IL}	$V_{IL} = 0.5V$		6.0	mA
I_{IH}	V_{IH} = logic high		.0	mA
I_I	$V_{IH} = 7V$		−2.2	mA

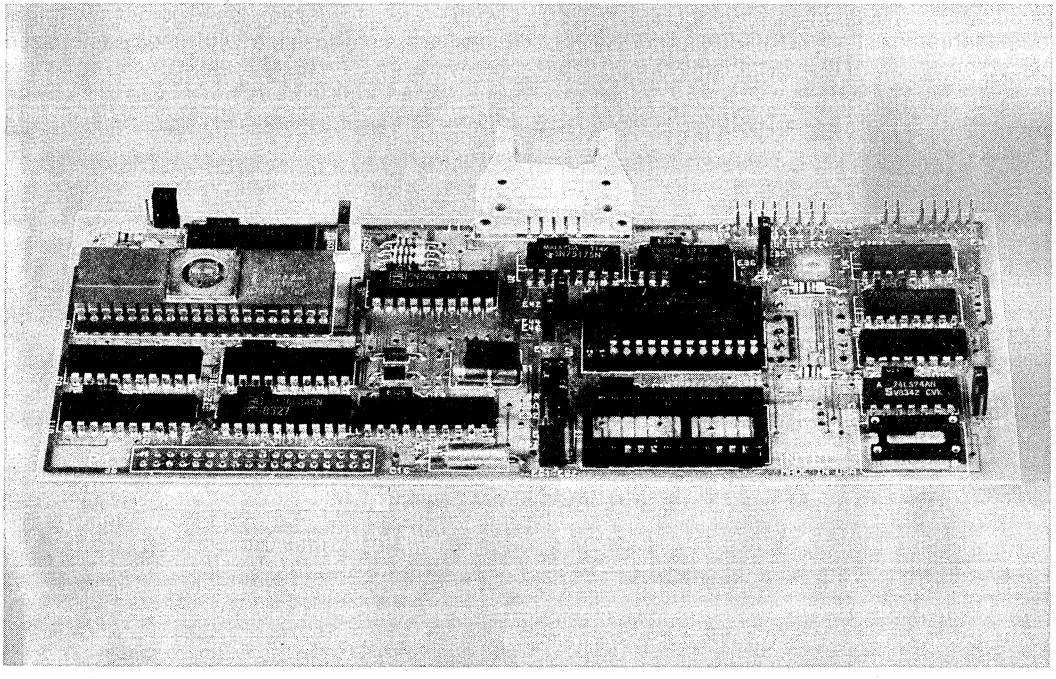
Ordering Information

Part Number	Description
iRCB 44/10	BITBUS Remote Controller board

iSBX™ 344
BITBUS™ CONTROLLER MULTIMODULE™ BOARD

- **High performance 12MHz 8044 controller**
- **Integral firmware including the iRMX™ 51 Executive optomized for real-time control applications**
- **Full BITBUS™ support**
- **2 28-pin JEDEC memory sites for user's control functions**
- **Low cost, double-wide iSBX™ BITBUS™ expansion MULTIMODULE™ board**
- **Power up diagnostics increase reliablity**

The iSBX™ 344 BITBUS™ Controller MULTIMODULE™ board is the BITBUS gateway to all Intel products that support the iSBX I/O Expansion Interface. Based on the highly integrated 8044 component (an 8 bit 8051 microcontroller and an SDLC controller on one chip) the iSBX 344 MULTIMODULE board extends the capability of other microprocessors via the BITBUS interconnect. With the other members of Intel's Distributed Control Modules (iDCM) family, the iSBX 344 MULTIMODULE board expands Intel's OEM microcomputer system capabilities to include distributed real-time control. Like all members of the iDCM family, the iSBX 344 MULTIMODULE board includes many features that make it well suited for industrial control applications such as: data acquisition and monitoring, process control, robotics, and machine control.



OPERATING ENVIRONMENT

Intel's Distributed Control Modules (iDCM) product family contains the building blocks to implement real-time distributed control applications. The iDCM family incorporates the BITBUS interconnect to provide standard high speed serial communication between microcontrollers. The two iDCM hardware products, the iSBX 344 MULTIMODULE board and the iRCB 44/10 BITBUS Remote Controller Board, communicate in an iDCM system via the BITBUS interconnect as shown in Figure 1.

As a member of the iDCM product line the iSBX 344 MULTIMODULE board fully supports the BITBUS microcontroller interconnect. Typically, the iSBX 344 MULTIMODULE board would be part of a node (master or slave) on the BITBUS interconnect in an iDCM system. As shown in Figure 2 the iSBX 344 MULTIMODULE board plugs into any iSBC® board with an iSBX connector.

The iSBX 344 MULTIMODULE board is the hardware interface between Intel's MULTIBUS® and iPDS™ ISIS environment and the BITBUS environment. With this interface the user can harness the capabilities of other Intel microprocessors eg: 80286, 80186, 8086 in a BITBUS/iDCM system or extend an existing MULTIBUS or iPDS ISIS-based system with the iDCM family.

MULTIBUS® and iPDS™ I/O Expansion

Typically, MULTIBUS iSBC boards have a maximum of two iSBX I/O expansion connectors. These connectors facilitate addition of one or two iSBX I/O MULTIMODULE boards with varying numbers of I/O lines. The iSBX 344 MULTIMODULE board increases the number of I/O lines that can be accommodated by a MULTIBUS system by at least an order of magnitude. The iSBX 344 MULTIMODULE board extends the I/O of Intel's Personal Development System (iPDS) or other systems products in a similar manner.

Extending BITBUS™/iDCM System Processing Capability

The iSBX 344 MULTIMODULE board allows utilization of other processors in a BITBUS/iDCM system to accomodate particular application requirements. The MULTIMODULE board is compatible with any iSBX connector so that any board having a compatible connector can potentially enhance system performance. Intel's iRMX 510 iDCM Support Package provides the software interface required for a variety of iSBC boards. The iSBC 186/03, 86/30, 286/10, and 188/48 boards are a few examples. Custom configurations are also possible with user customized software.

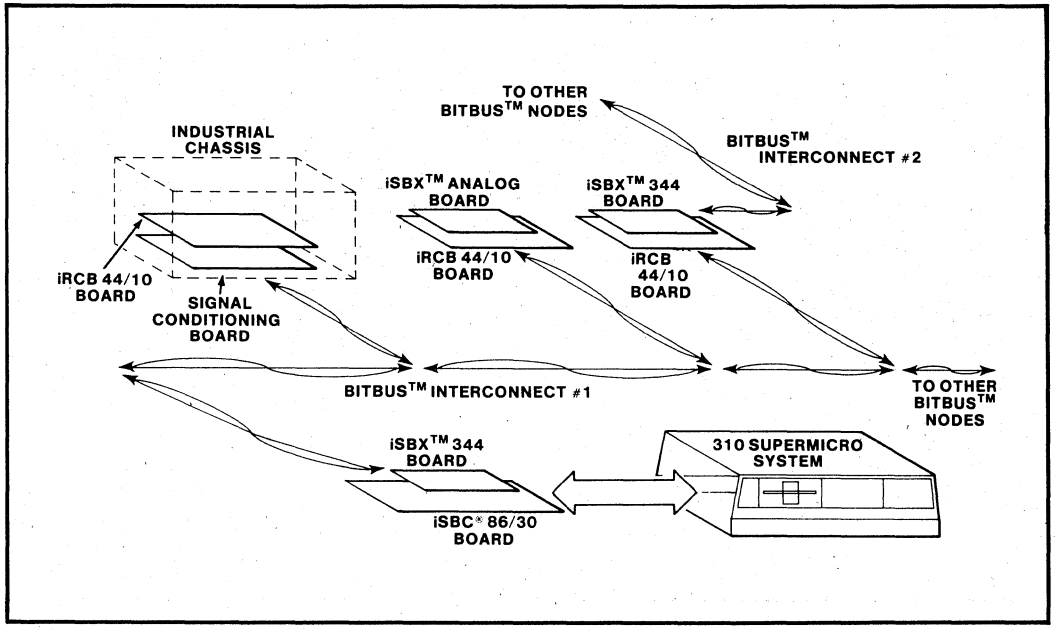


Figure 1. iDCM Operating Environment

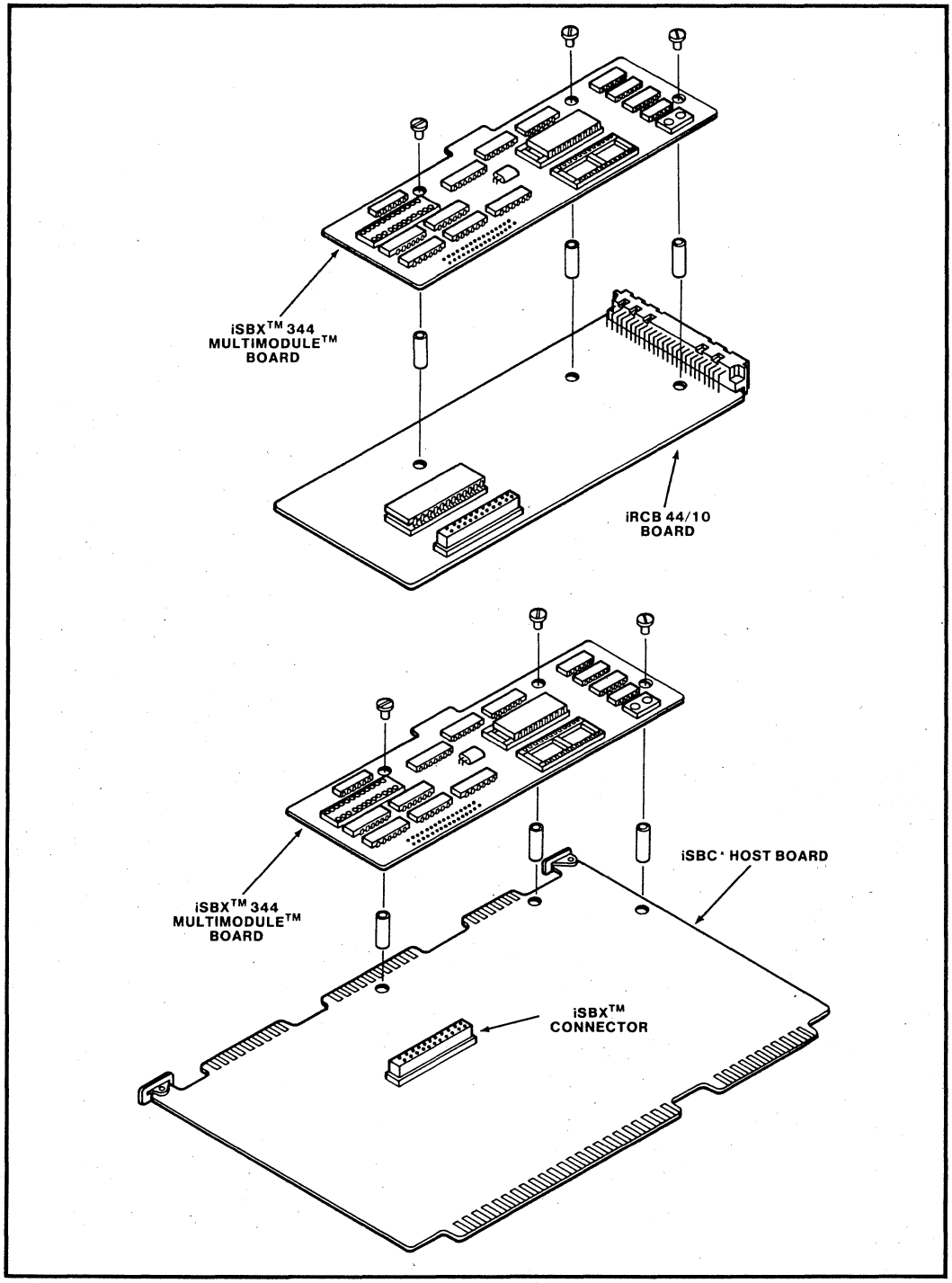


Figure 2. iSBX™ 344 Installation

ARCHITECTURE

Figure 3 illustrates the major functional blocks of the iSBX 344 MULTIMODULE board : iDCM controller, memory, BITBUS microcontroller interconnect, Byte FIFO interface, initialization and diagnostic logic.

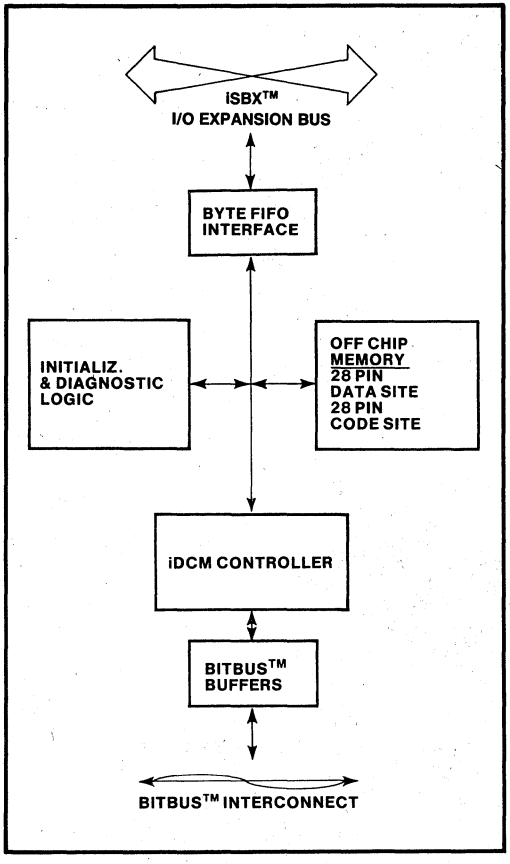


Figure 3. iSBX™ 344 Block Diagram

iDCM Controller

The heart of the iSBX 344 MULTIMODULE board's controlling and communication capability is the highly integrated 12 MHz 8044 microcontroller. The 8044 consists of the advanced 8-bit, 8051 microcontroller and a SDLC controller called the Serial Interface Unit (SIU). This dual processor architecture allows complex control and high speed communication to be realized cost effectively.

Another essential part of the iDCM controller is the integral firmware that resides on-chip to implement the BITBUS interface. In the operating environment of the iSBX 344 MULTIMODULE board, the 8044's SIU acts as a SDLC controller which offloads the on-chip 8051 microcontroller of communication tasks; freeing the 8051 to concentrate on real-time control.

The iDCM controller (8044 microcontroller and on-chip firmware) provides, in one package, a simple user interface, and high performance communications and control capabilities to efficiently and economically build a complex control system.

Memory

The iSBX 344 MULTIMODULE board memory consists of two sections: internal and external. Internal memory is located in the on-chip memory of the iDCM controller. The iRMX 51 Executive and the remaining iDCM controller firmware ration this resource. However, eight bytes of bit addressable internal memory are reserved for the user. Ample space is reserved for user programs and data in the iSBX 344 MULTIMODULE board external memory.

Two 28 pin JEDEC sites comprise the iSBX 344 MULTIMODULE board external memory. One site has been dedicated for data; the other for code. Table 1 lists the supported memory devices for each site. Intel's 2764, 27128, and 2817A are a few examples. The user may choose one of two memory configurations and specify different memory sizes by placing the proper jumpers at system initialization. The most flexible configuration option provides the user with access to the code site for program download or upload. This feature ensures expansion of an existing system is easily accomodated. For example, the addition of another conveyor to a material handling system would require adding another controller or controllers and changes to existing applications code and addition of new code.

Table 1. Supported Memory Devices

DEVICE	DATA SITE	CODE SITE
4K x 8-64K x 8 EPROM/ROM	NO	YES
2K x 8-32K x 8 SRAM	YES	YES
2K x 8-16K x 8 NVRAM and E2PROM	NO	YES

BITBUS™ Microcontroller Interconnect

The iSBX 344 MULTIMODULE board fully supports the BITBUS microcontroller interconnect. The BITBUS interconnect is a serial bus optimized for control applications. The interconnect supports both synchronous and self-clocked modes of operation. These modes of operation are selectable dependent on application requirements as are the transmission rates. Table 2 shows different combinations of modes of operations, transmission rates, and distances. The SDLC protocol, BITBUS message format, and compatibility with Intel's other software and hardware products comprise the remainder of this established architecture. These features contribute to BITBUS reliability and usefulness as a microcontroller interconnect.

The BITBUS connection consists of one or two differential pair(s) of wires. The BITBUS interface of the iSBX 344 MULTIMODULE board consists of a half-duplex RS 485 tranceiver and an optional clock source for the synchronous mode of operation.

Byte FIFO Interface

The Byte FIFO Interface on the iSBX 344 MULTIMODULE board implements the required hardware buffering between the iDCM controller and an extension. An extension is defined as a device attached to the iSBX I/O expansion interface on the iSBX 344 MULTIMODULE board. In an iDCM system, an example of an extension is an iSBC 86/30 board which may be considered the host board in a MULTIBUS system. When used with the software handlers in the iRMX 510 iDCM Support Package, implementation of this interface is complete.

For particular applications, the user may wish to develop a custom software interface to the extension or host board. On the iSBX 344 MULTIMODULE board side of the interface the iDCM firmware automatically accepts messages for the FIFO. No user code is required increasing the time available for application system development.

The Byte FIFO supports both byte and message transfer protocol in hardware via three register ports: data, command, and status. The extension side supports polled, interrupt, and limited DMA modes of operation (e.g. 80186 type DMA controllers).

Initalization and Diagnostic Logic

Like the other members of Intel's Distributed Control Modules (iDCM) product line, the iSBX 344 MULTIMODULE board includes many features which make it well suited for industrial control applications. Power up diagnostics is just one of these features. Diagnostics simplify system startup considerably, by immediately indicating an iDCM controller or external bus failure. The LEDs used for power up diagnostics are available for user diagnostics after power up as well as to further contribute to reliable operation of the system.

Initial iSBX 344 MULTIMODULE board parameters are set by positioning jumpers. The jumpers determine the BITBUS mode of operation: synchronous, self-clocked, transmission rate, and address of the iSBX module in the BITBUS system. This minimizes the number of spare boards to be stocked for multiple nodes, decreasing stocking inventory and cost.

INTEGRAL FIRMWARE

The iSBX BITBUS Controller MULTIMODULE board contains resident firmware located in the iDCM Controller. The on-chip firmware consists of: a pre-configured iRMX 51 Executive for user program development; a Remote Access and Control (RAC) function that enables user communication and control of different microcontrollers and I/O points; a communications gateway to

Table 2. BITBUS™ Microcontroller Interconnect Modes of Operation

	Speed Kb/s	Maximum Distance Between Repeaters M/ft	Maximum # Nodes Between Repeaters	Maximum # Repeaters
Synchronous	2400	30/100	28	0
Self Clocked	375	300/1000	28	2
	62.5	1200/4000	28	10

connect the BITBUS interconnect, iSBX bus, and iRMX 51 Executive tasks; and power up diagnostics.

The iRMX 51 Executive is an event-driven software manager that can respond to the needs of multiple tasks. This real-time multitasking executive provides: task management, timing, interrupt handling, and message passing services. Table 3 shows the iRMX 51 user interfaces. Both the executive and the communications gateway allow for the addition of up to seven user tasks at each node while making BITBUS operations transparent.

The Remote Access and Control Function is a special purpose task that allows the user to transfer commands and program variables to remote BITBUS controllers, obtain the status of a remote I/O line(s), or reverse the state of a remote I/O line. Table 4 provides a complete listing of the RAC services. No user code need be written to use this function. Power up tests provide a quick diagnostic service.

The services provided by the iSBX 344 MULTIMODULE board integral firmware simplify the development and implementation of complex real-time control application systems. All iDCM hardware products contain integral firmware thus supplying the user with a total system solution .

DEVELOPMENT ENVIRONMENT

Intel provides a complete development environment for the iSBX 344 MULTIMODULE board. Software development support consists of : the 8051 Software Development Package, and the iRMX 510 iDCM Support Package. The 8051 Software Development Package provides the RL 51 Linker and Relocator Program, and ASM 51. PL/M 51 is also available. The iRMX 510 Support Package includes the iDCM Controller firmware files on diskette as well as iRMX 51 libraries. Hardware tools consist of the IN-Circuit Emulator (ICE-44), Intel's Portable Development System (iPDS with EMV-44), and Intellec Series II or III Development Systems.

Table 3. iRMX™ 51 Interfaces

COMMAND	DESCRIPTION
RQ SEND MESSAGE	Sends a message (a command from the BITBUS master, a response from a slave, or a simple message between tasks on the same BITBUS component) to another task.
RQ WAIT	Waits for an interrupt, and event time-out, a message, or any combination of the three.
RQ CREATE TASK	Causes a new sequence of code to be run as an iRMX 51 task with a specific function identification code and priority.
RQ DELETE TASK	Stops the specified task and removes it from all execution lists.
RQ ALLOCATE	Allocates a fixed-length buffer from the on-chip, scratch-pad RAM for general use, or, in BITBUS applications, for a BITBUS message buffer.
RQ DEALLOCATE	Returns an on-chip buffer to the system.
RQ SET INTERVAL	Set the time interval to be used as a separate event-timer for the task.
RQ ENABLE INTERRUPT	Allow external interrupts to signal the microcontroller.
RQ DISSABLE INTERRUPT	Stops all external interrupts from signaling the microcontroller.
RQ GET FUNCTION ID	Provides a list of the 8 function identification codes representing the tasks currently operating on the microcontroller.

Table 4. RAC Services

COMMAND	DESCRIPTION
READ I/O	Read external I/O location. Return result in reply message.
WRITE I/O	Write byte to external I/O location.
UPDATE I/O	Write byte to, then read byte from external I/O location. Return result in reply message.
OR I/O	OR data with contents of external I/O location. Return OR'd value.
AND I/O	AND data with contents of external I/O location. Return AND'd value.
XOR I/O	XOR data with contents of external I/O location. Return XOR'd value.
READ INTERNAL MEMORY	Read contents of internal memory location. Return result in reply message.
WRITE INTERNAL MEMORY	Write data to internal memory location.
DOWNLOAD EXTERNAL MEMORY	Write data starting at external memory location.
UPLOAD EXTERNAL MEMORY	Read data starting at external memory location. Return result in reply message.
GET FUNCTIONS	Provides a list of the 8 function identification codes representing the tasks currently operating on the microcontroller.
CREATE TASK	Causes a new sequence of code to be run as in the iRMX™ 51 interface.
DELETE TASK	Stops the specified task and removes it from all execution lists as in the iRMX™ 51 interface.
RAC PROTECT	Suspends or resumes RAC Services.
RESET DEVICE	Returns device software to original state at initialization.

NOTES:
Internal memory locations are included in the 192 bytes of data RAM provided in the microcontroller. External memory refers memory outside the microcontroller — the 28-pin sockets of the iSBX 344 module and the iRCB 44/10 board. Each RAC Access Function may refer to 1, 2, 3, 4, 5, or 6 individual I/O or memory locations in a single command.

SPECIFICATIONS

Word Size

Instruction — 8 bit
Data — 8 bit

Processor Clock 12 MHz

Instruction Execution Times

1 μsec 60% instructions
2 μsec 40% instructions
4 μsec Multiply & Divide

Memory Capacity/Addressing

iDCM Controller — 64 K

Address Range

	Option 1	Option 2
External Memory		
Data	0000H-7FFFH	0000H-7FFFH
Code	1000H-0FFFFH	8000H-0FFEFH
Internal Memory		
Code	0000H-0FFFH	0000H-0FFFH

Terminations

Sockets provided on board for ¼ Watt 5% Carbon type resistors. Resistor value to match characteristic impedance of cable as closely as possible — 120 ohms or greater.

iDCM Controller (8044 + firmware) I/O Addressing as viewed from the 8044

FUNCTION	ADDRESS	READ	WRITE	BIT	COMMENTS
Data	FF00H	✓	✓		
Command	FF01H	✓	✓		Write sets command to extension — Read clears command from extension
Status					
-RFNF*	B3H	✓		✓	Also INT1 Input
-TFNE*	B2H	✓		✓	Also INT0 Input
-TCMD*	92H	✓		✓	
LED #1	90H	✓	✓	✓	
LED #2	91H	✓	✓	✓	
RDY/NE*	B4H	✓	✓	✓	
Node Address	FFFFH	✓			
Configuration	FFFEH	✓			

iSBX™ 344 MULTIMODULE™ board I/O Addressing as viewed from the iSBX™ 344 MULTIMODULE™ board

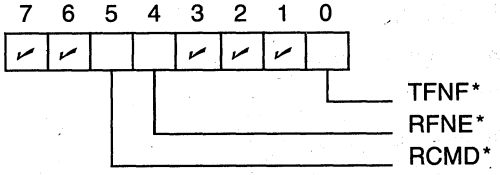
REGISTER FUNCTION	ADDRESS	COMMENTS
Data	Base	Read/Write
Command	Base + 1	Write sets command from extension Read clears command to extension
Status	Base + 2	Read Only

NOTES:
1. Base is determined by MCSO* on extension device

Interrupt/DMA Lines

SIGNAL	LOCATION	INTERFACE OPTION
RINT	MDRQ/MINT0	INT
TINT	MINT1	INT
RCMI	OPT0	INT OR DMA
RDRQ	MDRQ/MINT0	DMA
TDRQ	MINT1	DMA

Status Register Interface



Connector Options

10 Pin Plug

Flat Cable — 3M 3473-6010, TB Ansley 609-1001M, or equal
Discrete Wire — BERG 65846-007, ITT Cannon 121-7326-105, or equal

Pin Out

PIN	SIGNAL
1	+12V
2	+12V
3	GND
4	GND
5	DATA*
6	DATA
7	DCLK*/RTS*
8	DCLK/RTS
9	RGND
10	RGND

Electrical Characteristics

Interfaces

iSBX™ I/O expansion bus — supports the standard I/O Expansion Bus Specification with compliance level D8

Memory Sites — Both code and data sites support the electrical Universal Memory Site specification

BITBUS™ Interconnect — The iSBX 344 MULTIMODULE board supports the BITBUS Specification as follows:

Fully supported synchronous mode at 2.4 Mbits/sec and self clocked mode for 375 kbits/sec and 62.5 kbits/sec

The iSBX 344 MULTIMODULE board presents one standard load to the BITBUS bus

Message length of 18 bytes supported

RAC Function support as shown in Table 4

Power Requirements

.9A at +5V ± 5% iSBX™ 344 MULTIMODULE™ board only — memory NOT included

Physical Characteristics

Double-wide iSBX™ MULTIMODULE™ Form Factor

Dimensions

Height — 10.16 mm (0.4 in) maximum component height
Width — 63.5 mm (2.50 in)
Depth — 190.5 mm (7.50 in)
Weight — 113 gm (4 ounces)

Environmental Characteristics

Operating Temperature — 0°C to 55°C at 200 Linear Feet/Minute Air Velocity
Humidity — 90% non-condensing

Reference Manual (NOT Supplied)

146312 — Guide to Using the Distributed Control Modules

Ordering Information

Part Number	Description
iSBX 344	BITBUS Controller MULTIMODULE board

intel®

APPLICATION NOTE

AP-224

September 1984

The BITBUS™ Interconnect: From Flight Simulation To Process Control, It Simplifies Distributed Intelligence

SHANKER MUNSHANI
RICHARD MCALISTER
PETER MACWILLIAMS

Reprinted from Solutions July/August 1984

Order Number 280072-001

APPLICATION NOTE

The BITBUS™ Interconnect: From Flight Simulation To Process Control, It Simplifies Distributed Intelligence

By Shanker Munshani, Richard McAlister and Peter MacWilliams

A large portion of microcontroller applications demand distributed modes of operation. Physically, this distribution can stretch from a few meters to several kilometers. The environment of operation varies from a very peaceful electrical environment to a very variable industrialized environment.

To accommodate changing application needs and technological advances, designers need a *flexible* interconnect for such systems that causes minimal impact to performance. Compatibility and the implementation of standards are key. Adhering to a standard has several advantages: designers of equipment need not waste time defining and testing their own standard, and end-users are more comfortable if the manufacturer has followed an industry standard. At the same time, another important feature is the capability of handling reliable communication activity without impacting CPU performance.

Intel's Distributed Control Modules (DCM) family accomplishes such goals for distributed applications. DCM defines an interconnect architecture and consists of:

The BITBUS interconnect – Interconnect serial control bus
iSBX™ 344 – BITBUS controller multimodule board
iRCB 44/10 – BITBUS remote controller board
iRMX™ 51 – Real-time multitasking executive
iRMX™ 510 – DCM support package
8044AH – 8-bit microcontroller with on-chip serial communication support

This application note will explain the structure and function of the BITBUS interconnect and explore its use in aircraft simulation and chemical process control.

The BITBUS Interconnect: Rationale and Structure

To connect microcontrollers in a distributed application, two common approaches involve either building a custom interface and a custom cabling mechanism, or using other interfaces such as RS 232.

Yet, custom interfaces are faced with several disadvantages. They are generally very expensive, and a designer must design an interface in addition to designing the system. They also lack flexibility. For example, it is often impossible to add more input/output connections to a custom interface once it is implemented. Finally, custom interfaces pose problems for the end-user: they require considerable support; the cabling is generally cumbersome and slow; the distance over which they can be used is usually quite limited and their reliability may not be sufficient. Interfaces such as RS 232 are not an ideal alternative, either, because of the large amount of software support and cabling required.

The BITBUS interconnect avoids many of these problems. The BITBUS specification defines the data link protocol, message structure, protocol for a multitasking environment and a set of high-level commands for remote I/O access and application task control. This makes it very convenient to write high-level software interfaces. The BITBUS interconnect's high-level of definition means that the interface requirements can be implemented in silicon with minimal real estate at a low cost. This in turn reduces the complexity level.

The BITBUS in its simplest form is a pair of twisted wires. The BITBUS operates in a half duplex mode and can be used either in point-to-point operation or in a multi-drop environment. Figure 1 illustrates these two forms of connection. The BITBUS architecture supports a subset of the Synchronous Data Link Control (SDLC) protocol.

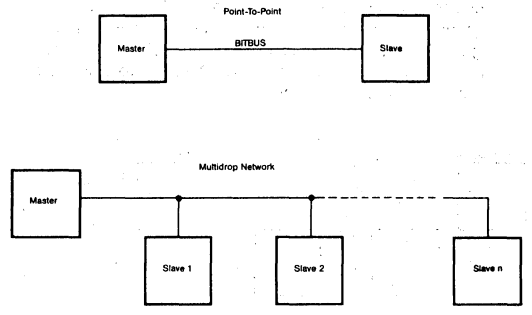


Figure 1.

There are three main objectives to be considered when using the BITBUS interconnect: speed of operation, distance over which communications has to take place, and number of nodes in the network. The BITBUS has two modes to meet these objectives: synchronous and self-clocked.

Synchronous mode

The synchronous mode is used for high speed operation. The distance over which this mode can be used is limited to 30 meters, and the number of modes in this set-up is restricted to 28 nodes. The speed of transmission in this mode is between 500 Kbits/sec to 2.4 Mbits/sec. To use this mode of operation, two pairs of twisted wires are required. One pair is used for the differential data clock signal (DCLK), while the other is used for the differential data signal (DATA). Figure 2 shows a typical synchronous mode interconnect.

Self-Clocked mode

In the self-clocked mode, as the name suggests, the clock is embedded in the data stream. In its simplest form, the

280072-001

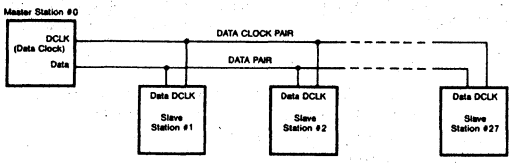


Figure 2. Synchronous Mode Network.

self-clocked mode requires just one pair of twisted wires. The speeds of operation in the self-clocked mode are 62.5 Kbits/sec and 375 Kbits/sec. The maximum distance of operation at 62.5 Kbits/sec is 1200 meters, and at 375 Kbits/sec is 300 meters. The maximum number of nodes in either case is 28. The self-clocked mode can be used to transmit over longer distances and to support more nodes by the use of repeaters. This, however, requires the use of an additional twisted pair of cables. This pair is used for Request To Send (RTS), which is the differential signal for transceiver control. The maximum number of repeaters allowed at 62.5 Kbits/sec are 10 and at 375 Kbits/sec are 2. Hence, at 62.5 Kbits/sec the distance over which the BITBUS link can be used is 13.2 kilometers, or 8.25 miles. The distance between the first node and the first repeater, the distance between two adjacent repeaters, and the distance between the last repeater and the last node are all called a segment. The maximum number of nodes permitted in any segment is 28, and the maximum number of nodes permitted in all the segments combined is 250. Figure 3 shows a self-clocked mode interconnect.

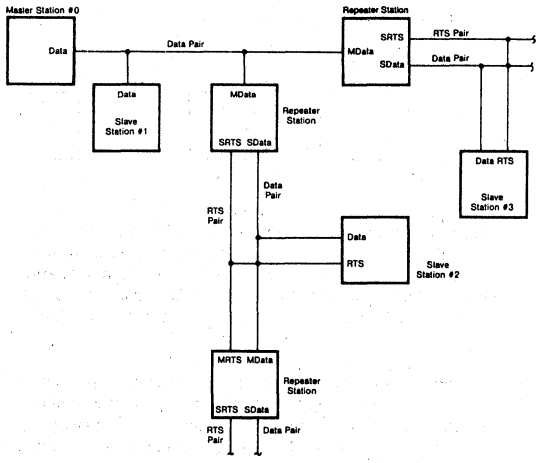


Figure 3. Self-Clocked Mode Interconnect.

The functions of the other parts of the BITBUS interconnect are described below:

iSBX 344 The iSBX 344 is a BITBUS controller multimodule board. This board can be used as either a master or a slave node in a BITBUS environment. This board has an iSBX connector and can be mounted on any iSBC board which has an iSBX connector and operates under any one of the following operating systems: iRMX 86, iRMX 286, iRMX 88, and ISIS-iPDS™ (Personal Development System). When the iSBX 344 multimodule board is used as a master node it is called a master extension, and when it is used as a slave node it is called a slave extension.

iRCB 44/10 The iRCB 44/10 board is a stand-alone BITBUS node. Unlike the iSBX 344 board, this board does not need a base board upon which to operate. The iRCB 44/10 board has a Eurocard single high-form factor and can be used as a stand-alone board. This board has 8 dedicated input lines and 16 programmable input/output lines.

iRMX 51 The iRMX 51 is a real-time, multitasking executive designed to monitor and control real-time events. A preconfigured version of the iRMX 51 Executive implements the BITBUS message format and provides all iRMX 51 facilities: task management, interrupt handling, and message passing.

iRMX 510 The iRMX 510 is a package of software aids to interface MULTIBUS® and iPDS ISIS systems to BITBUS systems in both run-time and development environments. It provides a simple software interface for iRMX 86, 88, 286 and iPDS ISIS operating systems compatibility. It provides a means for inexpensive remote control and communication in MULTIBUS-based systems.

8044AH The 8044AH with the DCM firmware provides the basic BITBUS interface. The 8044AH integrates a high performance 8-bit microcontroller, the Intel 8051 core, with an intelligent/high performance serial communication controller, called the Serial Interface Unit. The on-chip ROM can be used for the DCM firmware.

By virtue of these products, support for the BITBUS interconnect comes at various levels. The 8044AH chip with the DCM firmware provides the designer with the facility to integrate the BITBUS into the system at the very lowest level. Alternatively, the iSBX 344 multimodule board with an iSBX connector can be plugged into a system design at the highest level. Since the BITBUS interconnect is intelligent, it is capable of handling reliable communication activity with minimal interaction with the host processor.

Setting up a BITBUS Network

Figure 4 shows a typical BITBUS network. iPDS, Intel's Personal Development System, can be used as a master station to control the BITBUS network. The iPDS is a stand-alone development system with a CRT, a keyboard and a 5-1/4" floppy disk drive. The iSBX 344 board can act as a master extension on the iPDS base-processor board. This master station is capable of controlling up to 249 slave stations in a multi-drop fashion. The iSBX 344 is numbered as station #0 and is connected via the BITBUS to station 1, which is an iRCB 44/10 board. The BITBUS is then used to connect to slave station number 2, which is another iRCB 44/10 board. From here the BITBUS is routed to an iSBX 344 board mounted on an iSBC 86/30 board, a MULTIBUS-based board. This is termed as station #4. (Note there is no station #3; the station numbering does not have to follow a sequential order). The BITBUS then routes over to station #9, an iSBX 344 board on an iRCB 44/10 board. After this the BITBUS travels to another iRCB 44/10 board, station #10. From here the BITBUS goes to station #15, which is an iRCB 44/10 board with an analog multimodule

280072-001

board. Thus, there is one master station number 0, and six slave stations with the following numbers: 1, 2, 4, 9, 10, 15.

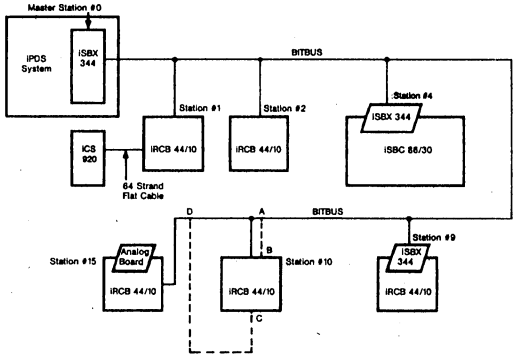


Figure 4.

If the distance from station 0 to 15 is less than 30 meters, this network can operate in either the synchronous mode or self-clocked mode. Assume the distance between stations 0 and 10 to be 200 meters, between stations 10 and 15 to be 250 meters and the speed required for the operation of the network to be 375 Kbit/sec. Since the maximum distance of a segment at 375 Kbits/sec is 300 meters, a repeater must be placed in the network. Since the iRCB 44/10 has on-board repeaters, station 10 could serve the function of a repeater. If this is the case, the BITBUS route then follows the direction of ABCD, as opposed to AD, as was the case in the previous example. Station 10 exists as a slave station and also as a repeater. Thus, the network has two segments, each less than 300 meters long. As a result, the network will work in asynchronous mode at 375 Kbits/s. It should be noted that stub lengths play an important role in a multi-drop network. Stub length is the distance from the drop point on the network to the node. Care should be taken to keep this as small as possible.

Creating a Task

Each individual station is now set up as an individual node. Since each BITBUS interconnect is an intelligent node, each node has its own tasks. Each station can have up to eight tasks. The Remote Access and Control (RAC) task is designated as task 0, so there can be seven more user-defined tasks. Using the same set-up as in figure 4, assume station 1 has only task 0 and no other user-defined tasks. This station could be used to perform any of the RAC Access or Control functions. A simple example could be to write a set of 1's to an output port and then flip this value to 0's. This could be achieved by using the EXT_IO_Write RAC function to write a set of 1's to an output port and then using the EXT_OI_XOR function to flip these bits. If an iCS 920 Digital Signal Conditioning board is connected to the output port of the iRCB 44/10 board, the LEDs (Light-Emitting Diodes) on that port of the iCS 920 board will turn on and off. If this task is run at a station, this will cause the LEDs to flash continuously. This demonstrates the simplicity of the I/O capabilities.

Message Structure The iRMX 51 Executive allows tasks to interface with one another via a simple message-passing facility.

Link: is a 2-byte field used by the executive.

Message_Length: is a byte value specifying the number of bytes in the message. This is 7 bytes of header information plus the number of bytes of user data. The maximum message size is configurable.

Message_Type: is a bit that determines whether this is an order message (=0) or a reply to a message (=1). If it is an order, the nucleus will use the consumer address as the destination. If it is a reply it will use the producer address as the destination.

Src_ext: is a bit value which indicates whether the sending task of an order message is located on an extension (=1) or on a device (=0).

Dest_ext: is a bit value which indicates whether the receiving task of an order message is located on an extension (=1) or on a device (=0).

Trk: is a bit field used during BITBUS transfer for tracking the message. Trk is set to 0 before sending an order message.

Station_address: For messages delivered locally (on the same chip), this field is 0. For messages delivered over a parallel interface only, this field is OFFH. For order types of messages, to be delivered from a master device or its extension to a slave device or its extension, this field is the SDLC station address of the slave device.

Source_task_id: is a byte value containing the task i.d. for the message originator. Upon reply, this value is interpreted as the reply destination.

Destination_task_id: is a byte value containing the task i.d. for the message destination. Upon reply, this value is interpreted as the reply source.

Command/response: is a byte field which is available for use by the sending and receiving tasks. It can be used for sending command or reply information. This field has pre-defined functions when communicating with the RAC function.

Message_information: is a user-defined field following the 7 bytes of message header information. For messages destined for the RAC task, this area has a fixed structure.

280072-001

Considering the example mentioned above to flash LEDs, the message sent and received would be (in Hex):

Message Sent	00	00	0B	40	01	00	0C	C2	FF	C0	FF
Message Received	XX	XX	0B	C0	01	00	00	C2	FF	C0	FF
									00		00

The first two bytes are the Link field. This field is reserved. The next byte specifies the message length, which is 7 bytes of header information plus 4 bytes of user-defined message. Therefore, the total message length is 11 bytes (i.e., OBH).

In the next byte, the first bit is set to 0 to indicate an order type and the second bit is set to 1, since the order message resides on an extension (the iSBX 344 is on an extension). The third bit is set to 0, since the task which receives the order message resides on a device (iRCB 44/10). The fourth bit is always set to zero before sending a message. The last four bits are reserved and set to 0's. This byte in binary is then equal to 01000000, i.e., 40H. In the received message the only field changed is the first bit, because now this bit is a reply and hence changes to 1. The received byte in binary is therefore equal to 11000000, i.e., COH.

The next byte defines the slave station address. Since the slave station address was 1, this field and its reply field are both 01.

The next byte is broken into two nibbles. Since the sending task and the receiving tasks were both RAC tasks (Task 0), this field is 00. (Note: this is not strictly the case for extensions).

Since the RAC order message has been generated, this field selects the RAC service for that message. The EXT_10_XOR RAC function has the value OCH. Thus the value at the ports defined will be Exclusive-ORed.

The last four bytes of the message follow the following format: address byte, followed by the data byte, followed by the address byte, and so on. The first byte (C2H) defines the address of the output port. The next byte (FFH) is the value written to this port (C2H). The next byte (C0H) is the address of another output port, followed by the byte value (FFH) written to this port (C0H). The received message has the same value in this field during the write operation. When the value is XOR the data field values change to 00H.

An Aircraft Application

Flight simulation uses the capabilities of the BITBUS interconnect. Figure 5 shows an implementation for flight simulation. As the figure demonstrates, flight simulation can be broken down into a block diagram level consisting of six sections, namely:

1) Pitch: This section is responsible for the vertical movements of the aircraft. The inputs required for this section

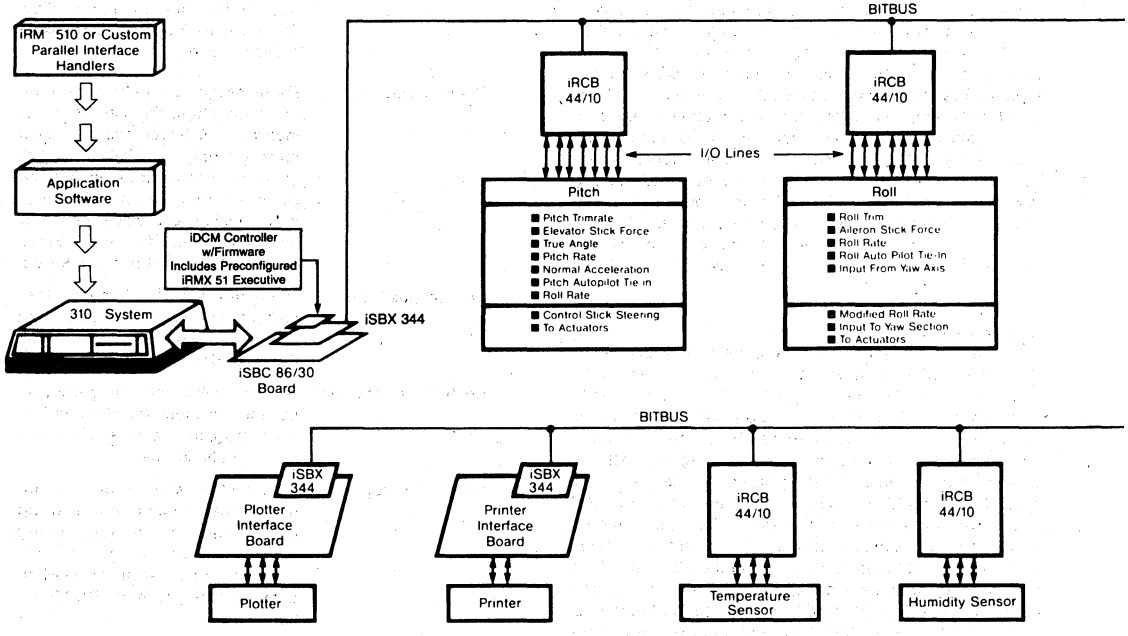


Figure 5.

280072-001

are pitch trim rate, elevator stick force, true angle, pitch rate, normal acceleration, pitch autopilot tie-in, and roll rate. The outputs of this section are: control stick steering and input to the actuators.

2) Roll: The Roll section is responsible for the rolling movement of the aircraft about its belly. The inputs required for this section are: roll trim, aileron stick force, roll rate, roll autopilot tie-in, and input from yaw axis. The outputs of this section are the modified roll rate, input to the yaw section, and input for the actuators.

3) Yaw: This section is responsible for the horizontal movements of the aircraft. The inputs for the yaw section are: yaw trim, rudder pedal force, yaw rate, lateral acceleration and yaw axis. The outputs of this section are input to the aileron rudder interconnect and input to the actuators.

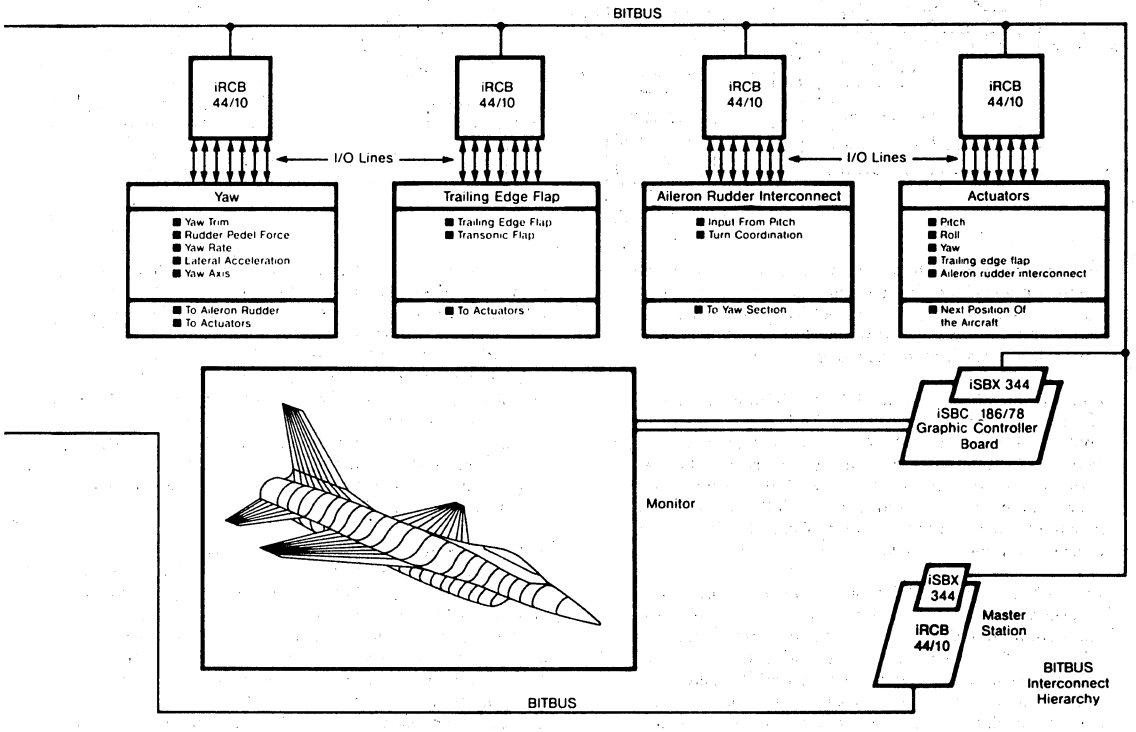
4) Trailing Edge Flap: The trailing edge flap section is responsible for the drag on the aircraft, mainly during take-off and landing. The inputs to this section are: trailing edge flap command, and transonic flap. The output of this section is input to the actuators.

5) Aileron Rudder Interconnect: When an aircraft rolls, its center of gravity shifts. Therefore, a force is required to counteract the gravitational force in order to keep the aircraft stable. This is achieved by the Aileron Rudder Interconnect. The inputs for this section are input from the pitch and turn coordination. The output of this section is input to the yaw section.

6) Actuators: Actuators are basically transducers which constantly monitor the aircraft. Their inputs are the various forces and factors currently acting on the aircraft and their outputs are the command signals for the next position of the aircraft.

In this example, Intel's 310 system is used as the master station. This is achieved by plugging an iSBX 344 board onto the iSBC 86/30 board inside the 310 system. The iSBX 344 board provides the BITBUS interconnect. Each section of the aircraft block diagram is controlled via an iRCB 44/10 board. (Depending on the device used to take the measurements and the accuracy desired, several iRCB 44/10s could be used in one section.) With this simple insertion, the BITBUS can be used to monitor each section. The actuators are also connected via the BITBUS, providing the control mechanism.

The BITBUS model monitors sections in the following manner: Each node (iRCB 44/10) has several tasks (a maximum of 8) residing on it. These tasks monitor the various parameters in each section. I/O ports on the iRCB 44/10 can be used to read the value of the different parameters in each section. They then perform the necessary computation and return the output parameters of that section to the master node. For example, in the "Pitch section", there would be a task to read the input port which is connected to a sensor monitoring the pitch trim rate. Similarly, there would be a task to compute the output of the pitch sec-



280072-001

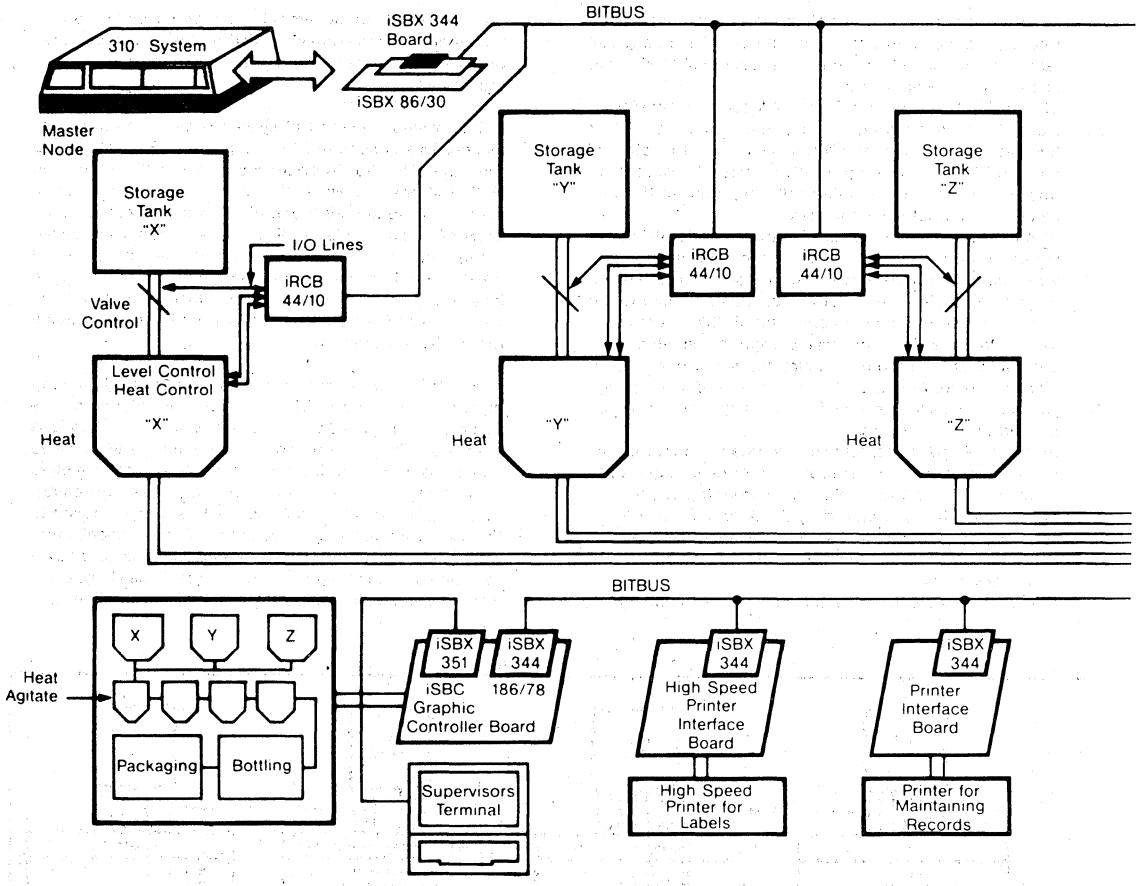


Figure 6.

tion and either write this value to an output port which controls a transducer, or send this value to the master so as to be used as an input to the actuators. In this manner, the BITBUS interconnect controls all the input and output parameters in each of the sections.

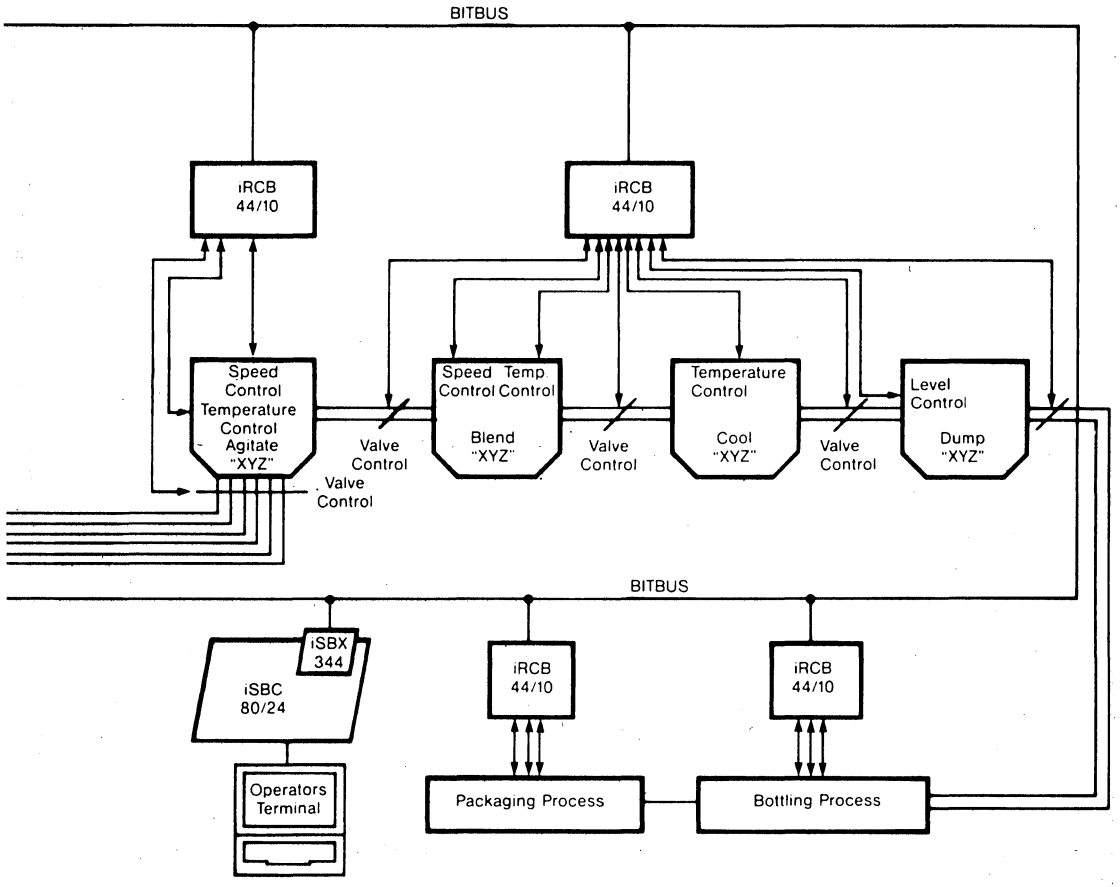
An iSBC 186/78 board is tied into the BITBUS network via an iSBX 344 board. The iSBC 186/78 board is a graphics controller board. The iSBX 344 board is just another node in the BITBUS network, and it helps in conveying the message from the master node to the iSBC 186/78 board. The master node sends messages to the iSBC 186/78 to display the simulations of the aircraft. The iSBC 186/87 then presents a graphical display on a CRT.

The BITBUS interconnect is defined to provide a high speed serial control bus for hierarchical systems. In many instances there are several slow devices or devices that do not need prime attention from the master node in a BITBUS network. This is a good reason to use the hierarchical facility of the BITBUS interconnect. In this example, the iSBX 344 multimodule residing on the iRCB 44/10 board uses this hierarchical interconnect. This iRCB 44/10 is now the master node for the four nodes that control the printer, the plotter, the temperature sensor, and the humidity sensor. The plotter logs the position of the aircraft at one minute intervals and the printer records the weather conditions every three minutes. The temperature and humidity sensors are tied into the I/O ports of the iRCB 44/10 nodes. These slave nodes continuously monitor the readings, and at the end of every 3-minute duration find the average value. These average values are then sent to the hierarchical master node, which in turn sends these values to the printer to log the values. The hierarchical master at the end of every minute receives the aircraft's position information from the main master and feeds this information to the plotter.

The BITBUS Approach to Process Control

Process control is another example where distributed intelligence is important. Figure 6 shows a simple process control flow. In this example, three chemicals, namely 'X', 'Y', and 'Z', are used to produce a product 'XYZ'. A 310 system is the master node. A slave node, in this case, an iRCB 44/10 board, is tied to each chemical unit at the start of the process. The I/O capabilities of this node control the flow of the chemical from the storage tank and the level of the chemical in the heating tank. Once the chemical reaches the required level in the heating tank, this node

280072-001



also closes the storage tank valve. After closing this valve, this node then turns the heater on in the tank and controls its temperature. At the end of the heating period, it opens the valve to the next tank.

Another node at the Agitate tank monitors the activities of this tank. This node controls the flow of the chemicals into the tank, the temperature of the chemicals, and the speed of the agitation motor.

Another node controls all the Blend, Cool and Dump stages. In the blend tank, the node controls the speed, temperature and the flow into and out of the tank. In the cooling tank, temperature and flow are controlled. The dump tank control monitors the level of the final product in this tank. If it reaches a near-full stage, it sends a message to the master station which then either stops the process momentarily or else diverts the action onto another dump tank. The BITBUS network then goes on to control the assembly line by controlling the bottling process and the packaging process. The same network is also used to log the packaged product information onto a line printer. Another node could be used to control a high-speed printer which would print labels with the batch number, the date of manufacture and the expiration date. If desired, an iSBX 344 node could be connected to an iSBC 186/78 board, which would run a color monitor in a supervisor's office, giving the supervisor a pictorial view of the entire manufacturing line. The BITBUS set up could also accommodate operators having a node at their benches to do any form of human interaction that is desired.

Conclusion

The BITBUS interconnect is capable of handling reliable communication activity without impacting CPU performance. It is a low-cost, high-performance approach that is easy to use. It does not require expensive cabling or special cables. It provides intelligent I/O capabilities. It has several speeds of operation in two modes and can be used over long distances at comparable speeds. The flexibility of the BITBUS interconnect makes it very attractive, since more slave nodes can be added with minimal effort. It is intended to be an important tool in an industrial environment, and, by virtue of its open architecture and standardized implementation, to continue to be of use as application needs evolve over time.

280072-001

intel®

ARTICLE REPRINT

AR-347

September 1984

BITBUS™ Promises to Standardize Control

MARK C. BUDZINSKI,
SR. TECHNICAL MARKETING ENGINEER

Order Number: 280053-001

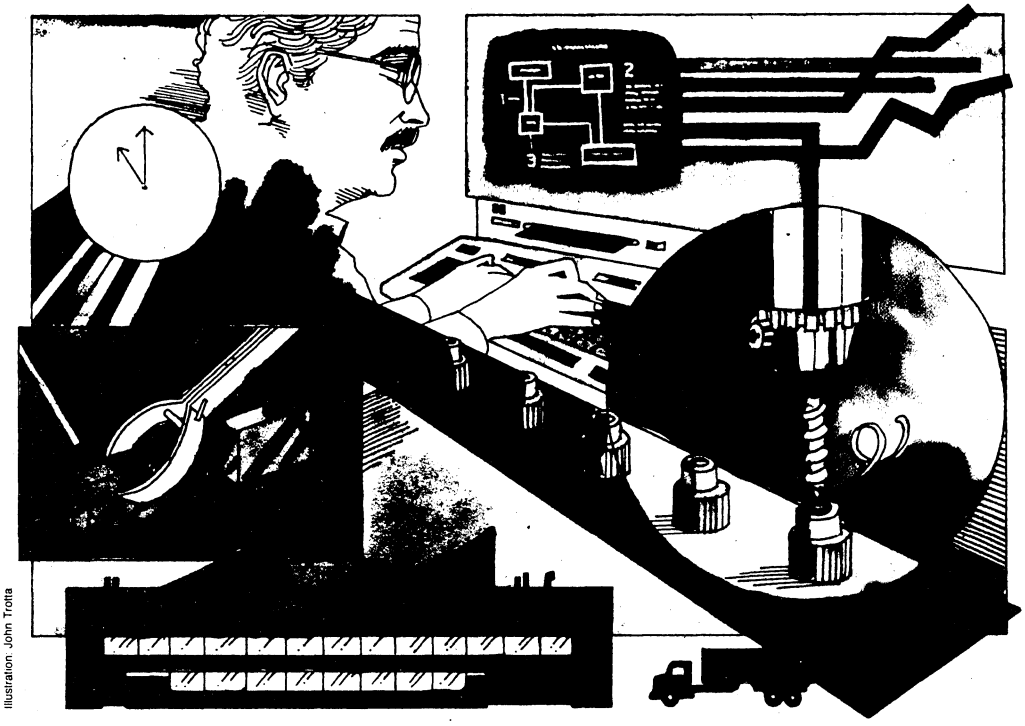


Illustration: John Trotta

Bitbus promises to standardize control

With a quartet of software and hardware modules, distributed control becomes much easier in the manufacturing environment.

The world of control systems is a rather confused one currently. Though the system designer has several excellent tools trying to meet the rigorous real-time requirements demanded by today's control applications and tomorrow's factory-automation desires, he is still put off by the lack of communications standards. Networks like Ethernet and IEEE-supported ones for token ring and bus find a niche in the manufacturing environment but do not specifically address problems residing in the control process. What is needed is a network that meets the needs of the control process and data com.

However, the Bitbus serial-network architecture represents a flexible solution to this problem (see "Specifications shoot for a flexible standard"). With just a simple twisted-pair, Bitbus may be configured in a multidrop set up with hardware- and software-interconnect modules and defined protocol support. The bus ranges from 30 meters to 10 kilometers in maximum length with speeds of 2.4 Mbaud to 62.5 kbaud, respectively; up to 250 nodes are possible with Bitbus (Fig. 1).

The key behind this standard is in the silicon of the 8044 microcontroller, which integrates an 8051 microcontroller and a serial interface unit that

Mark C. Buckfield, Sr. Technical Marketing Engineer
Intel Corp.
5200 NE Elam Young Pkwy
Hillsboro, Ore. 97123

Bitbus architecture

arbitrates control and interface functions, respectively. The 8051 can run closely to its 12-MHz clock rate because many babysitting functions, which in other systems are controlled by the main processor, are coordinated by the serial interface unit. Because the 8051 microcontroller will be placed locally at each control point, real-time control and task execution is now possible—a goal unattainable with more commonly used distributed architectures.

Control problems. Typically in control systems, it is desirable for a host computer to communicate with microcontrollers at remote locations. Bus-arbitration schemes like carrier-sense, multiple-access with collision detection are useful for local networks in many applications but are not suited for or even needed in the industrial environment. The overhead embedded in bus contention may thus be eliminated through a master-slave relationship between the host and remote microcontrollers—the master simply polls all of the microcontrollers.

There are a few applications that address the master-slave model: data acquisition, factory automation, and process control. Data acquisition usually deals with large amounts of input, and the input sites tend to have a limited amount of intelligence. Host-to-remote microcontoller transactions occur quickly so that all nodes may be polled often. The host computer can compile the required information and respond to the slave nodes with the appropriate acknowledgments or signals.

Factory automation requires more substantial remote intelligence, but a host computer still maintains overall system control. Readouts from I/O devices are generally dependent upon the conditions of their peers. For instance, a conveyer belt's speed is directly dependent on inputs from sensor devices located on the belt. As a result, communication must be both fast and reliable.

A process-control application usually involves the same peer interdependence as factory automation, but more wiring and local communications are required. Input devices can affect linked output devices without host intervention because a control system essentially resides at each remote node. The host acts as the system's global watchdog. Thus, communication must be fast, reliable, and cover significant distances.

Generally, control messages are short, concise statements that can be encoded in a few bytes. Remote controllers just idle until stimulated by the host that is controlling the system. The control message lets a microcontroller influence a process.

Data-acquisition messages originate differently. The host polls each microcontroller to send back temperature, pressure, or whatever type of information that it wants to examine. Again, the reply data can usually be encoded into a few bytes.

Because there may be many I/O points in a system, it is desirable to poll all of the microcontrollers as quickly as possible. For instance, process control would require more frequent polling than an environmental control system.

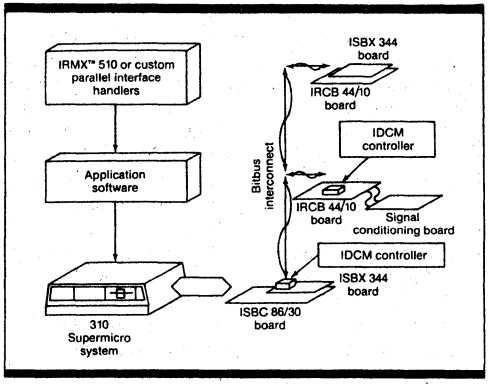
Transmission considerations. Transmitting and collating this data is not a trivial matter. Parallel buses provide the needed bandwidth but at a high price. In addition, the number of slots in a backplane is always limited. Serial buses are cheaper but suffer from the fact that RS-232-C is just a connection standard, not a protocol, and can offer only point-to-point communications.

The control environment presents fairly specific requirements to the more global field of data communications. Master-slave relationships, short messages, and many I/O points are typical of control systems. Unfortunately, there is no standard method available to bring all these elements together.

Bitbus arrives. Bitbus's architecture addresses many of these problems. At the physical level, RS-485 is utilized. This is a superset of the more popular RS-422 and provides multidrop support over a twisted pair. Logical states 0 and 1 are derived from the voltage across the pair. Any noise will raise the absolute potential, but the differential voltage potential will remain constant throughout the bus thus making the line fairly noise resistant. RS-485 is used instead of RS-422 because it allows more nodes over a given distance. Bitbus networks support 32 nodes in each 1200-m segment.

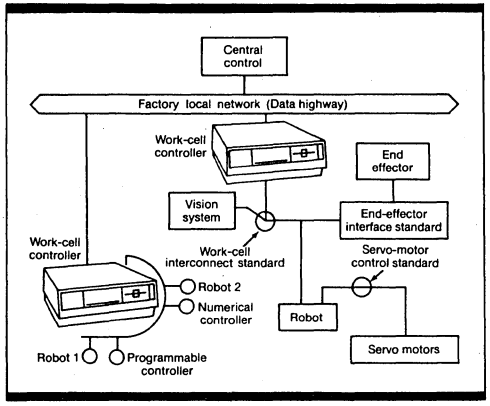
The data-link level is supported by a subset of IBM's synchronous-data-link-control (SDLC) protocol, which is well-suited for the control environment. First, it provides a header field for addressing the remote nodes in a system. Second, it provides excellent reliability through its cyclic-

1. The typical system configuration of Bitbus comprises iRMX 510 or custom parallel interface handlers, several iRCB 44/10 boards, a iSBX 344 board, and firmware containing the iRMX 51 executive. Bitbus links all these elements into a communicating multidrop network.



redundancy-check algorithm—the probability of passing an erroneous bit is less than 10^{-10}. Third, it provides frame-for-frame reliability by keeping track of outstanding frames, as well as maintaining an acknowledge mechanism. The protocol is well-accepted throughout the data-communications field, thus contributing to a potential standardization for control applications.

Most common data-communications packages, particularly those in the microcontroller area, end their support at this level. The Bitbus architecture goes an additional step and establishes end-to-end communications through a defined message header and efficient routing. The message header contains information that routes the message to the correct



2. Bitbus's architecture is quite useful in the factory where it can be used to link dissimilar devices like robots, programmable controllers, and vision systems. Because all the interface levels—from the physical to the protocol—are defined, interconnection is easy and straightforward.

node and in turn to the correct process running at that node. Message length is specified so that dynamic structures can be used to create messages. Maximum message size is currently 18 bytes and includes the header information.

Only 4 bytes out of the initial 18 are used for the message header. Because control applications require only short message lengths, typically, the header-field requirement does not weaken the network.

Bitbus is set up so that the master node keeps track of outstanding order messages that have to be polled. Thus, when an order is sent to a slave node, a reply is guaranteed in a subsequent poll. There are no wasted polls, which adds to system efficiency.

Distributed control modules. A quartet of software and hardware components support Bitbus communication and are called distributed control modules (DCMs). The two board-level products are the iSBX 344 and iRCB 44/10. The software components comprise the iRMX 51 real-time multitasking executive and the iRMX 510 Bitbus utility package.

The 344 is a single-height, double-wide multimodule board designed to interface with a Multibus-compatible central-processing-unit board. The 44/10 remote controller board serves by itself as a good solution for a stand-alone controller node in a system. Twenty-four lines of I/O are provided from the board in addition to Bitbus communications capabilities. Both boards are based on the 8044 microcontroller, which is composed of an 8051 CPU and a serial-interface unit; they provide 64-kbytes of both data and instruction memory. The software is provided on both boards in the form of preconfigured firmware.

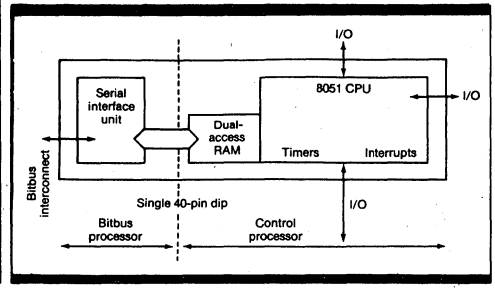
Designing a control system with these modules is very straightforward. Initially a host computer must be selected. For instance, the Intel system 310 running the iRMX 86 operating system is a good choice. It is easily implemented into a Bitbus system by adding just one iSBX 344 board.

The heart of the entire DCM scheme is the real-time control provided by the microcontroller. The iRMX 51 operating system offers an environment where eight tasks can run on the same microcontroller. It is an interrupt-driven executive that utilizes a convenient interface through system calls. It supports the Bitbus message format, allowing easy implementation into a distributed control system. Also, it is included as part of the firmware available with the DCM board-level products.

The iRMX 51 executive is an excellent development tool. The functions that have to be explicitly coded can be greatly reduced by making system calls. Once the system is up and running, the executive handles all interrupts, context switching, message passing, buffer management, and timer management transparently.

Real-time support is another characteristic of the iRMX 86 operating system, which is run at the host. A Bitbus driver runs under the operating system,

3. The heart of the Bitbus specification is the 8044 microcontroller. It combines the 8051 microprocessor and a serial interface unit on one chip so that interface demands and control needs are separated. This arrangement leads to quicker response times.



which allows message passing across the iSBX interface and on down the Bitbus network. This allows 8051 messages to be transmitted to and from the iRMX 86 host conveniently.

Routing operations. The 4-byte message header has the routing and control information that travels with every message. The first byte represents the length of the transmitted message. This overhead byte quickly pays for itself when smaller messages are transmitted—statically defined space is not wasted.

The next field provides a level of routing information. Four flags are set: message type, source extended, destination extended, and track bit. Message type specifies whether the message is going from master to slave or vice versa. Source- and destination-extended bits allow a level of routing across another bus interface (the iSBX bus, in the DCM case). If a non-Bitbus interface is required in the system, the extended bit is set so it is clear that the message has to be routed across another interface. The track bit is used to keep messages going forward across multiple-extended interfaces.

The third byte in the header specifies the station address, which is the SDLC address, of the destination node. The fourth byte identifies the 8051 sending and receiving task. Under the iRMX 51 executive there are up to eight tasks running on the microcontroller.

The routing strategy is directly related to the defined message format. First, a node is specified to receive a message. A task is then sent a message from the node and, if necessary, an extended interface is configured. If the message is an order, it will go to the destination task; if it is a reply, it will come back to the appropriate task at the master.

iRMX 51 interfacing. Interfacing with the 51 executive is quite easy. The executive is small, having only 12 system calls. Available functions include, send message; wait-on-message, timeouts, interrupts, or intervals to end; allocate buffer space to place a pending message; create and delete tasks dynamically set intervals; and disable interrupts.

In sending a message, a structure is created that is compiled with the Bitbus message format; a pointer is passed to it via the "send message" system call. All routing is handled by the system in conjunction with the DCM software. The wait system call is very much like the "go-to-sleep" call found in other executives. The task waits for a specific event to occur; it "wakes up" once the event happens. Task priorities are such that no tasks of equal or less priority than the running task can preempt it. Thus, if a task is waiting on a message and also is of highest priority, it will run immediately upon receiving the message.

Allocate system calls permit a user to easily send messages across the Bitbus network. Upon allocating a buffer for the message, it is simply sent—the system expects to see it in the buffer and then sends the message. Create-and-delete-task system calls provide the system with the capability to ring up code during runtime. In addition, a task may be deleted during runtime in response to a condition like an error.

The iRMX 51 executive is designed to run on 8044 processors as well as 8051s. As a result, firmware is provided in the board-level products to handle communication over the Bitbus network transparently. When a communication interrupt occurs, the DCM code flags it and sends a message across either the Bitbus or the iSBX bus. Since Bitbus accommodates the iRMX 51 executive, the task is straightforward. The DCM software is implemented as Task 0 under the iRMX operating system.

In addition, the DCM code provides a user interface (destination Task 0) that can send redefined messages. These messages allow reading and writing to certain memory locations. The I/O locations on the iRCB 44/10 board are memory-mapped in these same locations. Thus the I/O ports of a remote controller can be accessed without writing any code at that node.

In addition, a task may be created at a remote

Specifications shoot for a flexible standard

Though bus and network products for industrial applications abound, few if any can claim to be inexpensive and sufficiently defined so that any two implementations of such a product can be linked easily. But with Bitbus, everything from the electrical interface to the operating system is specified. As a result, compatibility between dissimilar products adopting the standard is ensured. Manufacturers such as Mitsubishi, Westinghouse, and Unimation have already expressed support for this specification.

The Bitbus interconnect method was developed to use a range of industry-standard approaches for simplicity of design and use. The Bitbus physical connection uses inexpensive twisted-pair wiring (2 pairs and 10-conductor flat ribbon are also specified) and the RS-485-compatible electrical characteristics. Besides the SDLC protocol, Intel's open-system software interfaces for the iSBX, Bitbus drivers, and intertask message passing are included. This setup is easy to configure and extremely flexible.

For instance, the interconnect provides for three different data-transfer rates and three maximum distances. The highest speed is 2.4 Mbaud over a four-wire cable over a maximum distance of 30 meters for up to 28 Bitbus nodes. Next comes 375 kbaud on two-wire cable, with a maximum distance of 300 meters between up to two repeaters. A third speed, 62.5 kbaud, permits a two-wire cable length of 1,200 meters between repeaters, with up to 10 repeaters between nodes. Up to 28 Bitbus nodes may be connected between repeaters in the latter two modes.

Bitbus architecture

location through the same interface. This feature lets task code be downloaded to a remote node and be created in real time from the master node. The task can later be deleted from the master by sending a similar message to Task 0.

Though the software flexibility contributes much to the desirability of Bitbus, the 8044 microcontroller (Fig. 3) is the heart of the system. It consists of an 8051 microcontroller with an SDLC controller, which share a dual-port RAM area. The 8051 section controls processes without the overhead of communications responsibilities. The SDLC controller or the serial interface unit (SIU), can generate frame acknowledgments, maintain frame-sequence counters, and other such things in hardware without the assistance of the main processor.

Because of this splitting of chores, Bitbus speed is relatively high. The slave end never has to interrupt the processor unless it has a valid message for input. All negative acknowledgments are transparent to the 8051 portion of the chip.

Message buffers, task stack space, iRMX 51 variables, and register banks sit in the dual-port RAM area, which is 192 bytes—sufficient for most applications. The system is optimized to a certain extent so that when an interrupt occurs, no context switch to off-chip RAM takes place unless the running task is preempted. If a running task continues to run after an interrupt occurs, there is no context switch to external memory.

iRCB 44/10 interfacing. The iRCB 44/10 boards have a single-width Eurocard form-factor with a 64-pin DIN connector. Power, ground signals, the Bitbus interface, and 24 I/O lines are physically connected to the DIN plug. The I/O lines are not optically isolated or conditioned in any fashion but provide simple binary 0s and 1s. To interface the I/O lines with real-world devices requires an intermediate interface like an optical isolator. Analog signaling is possible when an iSBX multimodule board is included.

The beauty of the I/O-line setup is not the direct-indirect interface to real devices, but their co-location with the controller board and their memory-mapped addressing that can be accessed from both the remote node as well as the host. Because the board is designed to stand alone, no backplane is necessary.

Development of application code may proceed with either assembly language or PL/M 51, which is a more sophisticated high-level language.□

intel®

ARTICLE REPRINT

AR-351

September 1984

Intel's BITBUS™ Microcontroller Interconnect

PETER WOLOCHOW
PRODUCT MARKETING MANAGER

Reprinted by permission from June 1984 Robotics Age Magazine, Robotics Age, Inc.

Order Number:280069-001

INTEL'S BITBUS MICROCONTROLLER INTERCONNECT

A Modern Method of Robot Communication

Peter I. Wolochow
Intel Corporation
5200 NE Elam Young Parkway
Hillsboro, OR 97124

In February 1984, Intel introduced a new bus communication architecture aimed at enhancing microcontroller-based applications. This article describes Intel's Bitbus—an interconnect scheme specifically designed to match the needs of high-performance, cost-conscious microcontroller applications. The Bitbus, along with the accompanying Distributed Control Module (iDCM™) family, provides the latest steps toward making the best use of VLSI technology in control applications.

BITBUS USES

Microcontrollers are a driving force in modernizing mechanical and electrical systems. They have replaced relays, wheels, and gears in applications ranging from automated manufacturing to process control. By incorporating at least one microcontroller, the total system cost of most control-oriented applications can be lowered while improving performance and leaving room for useful options. However, without an industry-wide standard on which to rely, most microcontroller applications lack a simple connection to other microcontrollers and control equipment.

Through advances in silicon technology, control systems have already made many evolutionary changes. Early systems relied on relay sequencers and simple alarm indicators. The PDP-8™ from Digital Equipment Corp. provided the first commonly used tool for coordinating many real-time controls. Microprocessors provided a simple way to reduce computer costs, but did not change system architectures until standard buses became popular. Industry standard bus architectures such as the Multibus® and STD-Bus™ have allowed designers to divide control tasks between many processors while taking advantage of standard modules from many vendors.

Microcontrollers have paved the road for the next step—distributed control. Most control-oriented systems distribute control functions to minimize system cost while improving system performance, responsiveness, and reliability. There are several ways to distribute control: a parallel bus structure, a simple set of control signals on individual control lines, a serial communications link, or custom technology. Each of these solutions requires considerable design effort, and often results in a performance-limiting interconnect matched to current applications but ill-suited for expansion or connection to a different control system.

Applications best suited for the Bitbus interconnect include robotics, numerically-controlled machines, process control, security systems, environmental control, and other distributed control and data collection systems. These applications typically use multiple controllers to physically distribute control, to improve system performance and reliability, and to reduce total system and maintenance cost. Existing data networks such as Ethernet, Token Bus, and various custom technologies are useful for transferring large data blocks at high speed, but also at a relatively high cost. What is still missing is a low-cost bus for local control environments. Combining the strengths of existing hardware and protocol standards with complete firmware and software support, the Bitbus interconnect provides a simple, standard technology for connecting distributed controllers.

BITBUS CONFIGURATIONS

The Bitbus microcontroller interconnect can connect a single master controller to a number of local or remote slave controllers. A multidrop configuration connects controllers (8044s) to a common bus. The single-chip *8044 microcontroller* contains two functional elements: an 8051 processor and an SDLC Serial Interface Unit (SIU). The onboard processor contains 4 Kbytes of read-only memory, 192 bytes of read/write memory, clock, timers, interrupt controller, and memory expansion bus. In Bitbus applications, the read-only memory is filled with special firmware routines that support the message-passing protocol, interact with user application tasks, and perform a series of power-up self-diagnostics.

Integral firmware allows each Bitbus controller to act as either a master or slave.

ROBOTICS AGE June 1984

RAC Function Commands

Every member of the 8044 family is programmed for Bitbus operation by including the iRMX 51 Executive and one system task, called the Remote Access and Control (RAC) function. The system task, responsible for managing the interface, provides a number of utilities to ensure that all Bitbus controllers can communicate with each other. The utilities also allow a master node to interact with slave controllers without having to write any 8051 code. The RAC function provides ten commands that let a remote master access local resource and status information and five commands specifically designed for intelligent remote control. Tables 3 and 4 describe the available access and control RAC functions.

The control functions can be used for higher-level utilities. For example, the Bitbus master can determine the existence of a special service task at a remote slave, and download programs depending on high-level system requirements or environmental influences such as service options or power failures. This test-and-program function could be accomplished by sending a GET_FUNCTION_IDS command and sending a group of WRITE_EXTERNAL_MEMORY messages to download an appropriate program, and then starting the program by sending a CREATE_TASK command.

Table 3. Remote Access and Control (RAC) Access Functions

COMMAND	OPERATION
READ I/O	Read external I/O location. Return result in reply message.
WRITE I/O	Write byte to external I/O location.
UPDATE I/O	Write byte to, then read byte from external I/O location. Return result in reply message.
OR I/O	OR data with contents of external I/O location. Return OR'd value.
AND I/O	AND data with contents of external I/O locaton. Return AND'd value.
XOR I/O	XOR data with contents of external I/O location. Return XOR'd value.
READ INTERNAL MEMORY	Read contents of internal memory location. Return result in reply message.
WRITE INTERNAL MEMORY	Write data to internal memory location.
DOWNLOAD EXTERNAL MEMORY	Write data starting at external memory location.
UPLOAD EXTERNAL MEMORY	Read data starting at external memory location. Return result in reply message.

NOTES:
Internal memory locations are included in the 192 bytes of data read/write memory provided in the microcontroller. External memory refers memory outside the microcontroller—the 28-pin sockets of the iSBX 344 module and the iRCB 44/10 board. Each RAC Access Function may refer to 1, 2, 3, 4, 5 or 6 individual I/O or memory locations in a single command.

Table 4. RAC Control Functions

COMMAND	OPERATION
GET FUNCTION IDS	Execute iRMX 51 GET FUNCTION IDS command. Return resulting list in reply message.
CREATE TASK	Execute iRMX 51 CREATE TASK command using the specified Task Descriptor. Return resulting status in reply message.
DELETE TASK	Execute iRMS 51 DELETE TASK primitive using specified Task Identification. Return resulting status in reply message.
RAC PROTECT	Suspend or Resume Remote Access types of service. Return resulting status in reply message.
RESET STATION	Jump to initial code reset address. **NO REPLY IS RETURNED.**

master and slave starts with an exchange of U-frames to synchronize the frame sequence counters and other controls. The data exchange is accomplished with the I-frame and S-frame. The I-frame contains data for a slave or a response from a slave. S-frames are used to acknowledge data receipt or to poll a slave for data.

Bitbus Messages. The *user data* contained in the I-frames conform to a standard Bitbus message format. All messages contain a five-byte header describing the source and destination, along with other status and control information. Messages sent to, or from, the standard Bitbus firmware (see the RAC Function Commands text box) may also contain special information to perform common I/O operations. Up to 13 user data bytes may be transmitted at one time.

Bitbus controllers *require* each message to be answered—not just acknowledged. The SIU provides an SDLC acknowledgement for each message. Only the application task to which the message is sent can send a meaningful reply. The Bitbus master enforces this rule by continuing to poll a slave for its reply to each transmitted message. Separating the replies and acknowledgements helps give Bitbus systems additional performance by freeing the control link while a reply is generated.

Figure 1 shows how several overlapping conversations can occur between the Bitbus master and slaves. The Bitbus master maximizes throughput by taking advantage of each slave's SIU ability to immediately acknowledge a message, without interrupting the 8051 processor. This removes the need to tie up the bus while waiting for the slave to calculate a response. The master can send commands to a controller and come back for the response later.

Bitbus traffic patterns and rules reflect the primary Bitbus purpose—control. The master controls all message traffic and initiates all messages. Slaves answer each message with either an immediate acknowledgement to confirm correct recognition, or with a response message associated with a previously received command. Whereas many SDLC-based systems suffer from large and unpredictable delays between a poll and a slave's acknowledgement, Bitbus slave controllers use the SIU to send an immediate answer. Because the response time is always short and predictable, no Bitbus bandwidth is wasted.

The master/slave relationship can be changed in real time. This provides a simple method for allowing backup master controllers. For example, you could program a backup master to wait for a poll from the primary master every second. If the master missed several polls, the backup could take control and switch itself from a slave to a master.

TRANSFER PROTOCOL AND MESSAGE FORMAT

The Bitbus interconnect is based on the SDLC (IBM's Synchronous Data Link Control) standard supported by the 8044. SDLC is commonly used by many vendors concerned about data integrity and interface standards. Since the SDLC protocol has limited overhead and a built-in data security and acknowledgement protocol, it is ideally suited for reliable transmission of short, control-oriented messages. To ensure a workable standard interface between Bitbus systems, additional protocol standards are included in the published Bitbus specifications.

The 8044 Bitbus microcontroller supports a large subset of the standard SDLC protocol. The 8044 manages SDLC traffic in Auto and Non-Auto modes with a minimum of interruption to the 8051 half of the microcontroller. In Auto mode (used by all Bitbus slaves), all SDLC functions are managed automatically with minimal effect on controller performance. In the Non-Auto mode (used by the Bitbus master), the controller can initiate transmissions and polls to slaves as well as process parts of the protocol not managed by the SIU (such as responding directly to each message and checking other status conditions). The 8044 component automatically manages all SDLC frame control, sequencing, and transmission procedures. In Bitbus applications, the 8044 also provides the message formats and sequence checks needed to guarantee proper delivery of critical control signals.

SDLC Protocol. SDLC is a bit-oriented data-link control protocol that defines a specific structure for each type of data and control exchange. As shown in table 1, each transmission type (I-frame, S-frame, and U-frame) is divided into identifiable fields. Each field contains one or more bytes of data and/or control information to accomplish the corresponding function.

Normal transmission between an SDLC

Transmission Modes

The Bitbus operates in either of two transmission modes: synchronous or self-clocking. In the synchronous mode, an external clock provides a data clock to transmit data at rates between 375 K and 2.4 M bits per second. In the self-clocking mode, the clock is derived from transition in the data using the NRZI (Non-Return to Zero, Inverted) encoding technique. Characteristics of each mode are shown below:

Synchronous Mode:
2.4 bits per second with external clock
Maximum of 30 m distance
Maximum of 28 Bitbus nodes

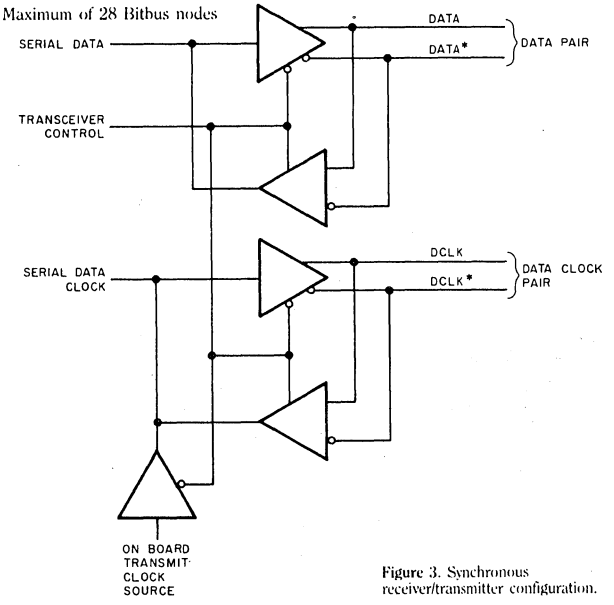


Figure 3. Synchronous receiver/transmitter configuration.

Self-Clocking Mode:
375K bits per second NRZI encoding:
Maximum of 300 m between repeaters (total limit 900 m)
Maximum of 28 Bitbus nodes between repeaters (limit 250)
Maximum of 2 repeaters between the master and any slave

62.5K bits per second NRZI encoding:
Maximum of 1200 m between repeaters (total limit 4800 km)
Maximum of 28 Bitbus nodes between repeaters (limit 250)
Maximum of 10 repeaters between the master and any slave

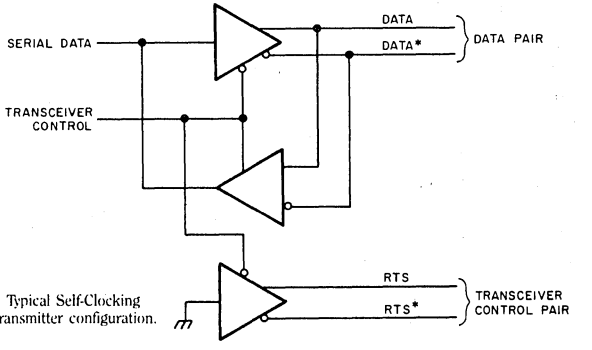
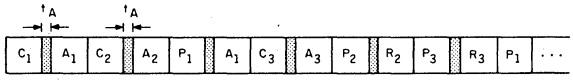


Figure 4. Typical Self-Clocking receiver/transmitter configuration.

Table 1. Synchronous Data Link Control (SDLC) Frame Formats.

FRAME-TYPE	FORMAT	FUNCTION
I-frame	F A C—user data—FCS F	Information Transfer
S-frame	F A C FCS F	Supervisory Control
U-frame	F A C FCS F	Receiver/Transmitter Synchronization

NOTES:
'F' refers to the SDLC Flag byte; 'A' refers to the slave station's address; 'C' refers to the control field identifying the frame-type and other control parameters; 'FCS' refers to the frame check sequence (a 16-bit CRC calculated on all frame contents except the flags).



Where:

$C_1 \ldots C_n$=
Command messages sent by the master to the identified slave.

$A_1 \ldots A_n$=
Acknowledgements made by identified slave to message from master.

A_m=
Acknowledgement made by the master to a message from a slave.

$R_1 \ldots R_n$=
Response message sent by identified slave to the master. The master acknowledges these responses with the next appropriate poll or command to the identified slave.

$P_1 \ldots P_n$=
Poll sent by the master to identified slave asking for response message.

t_A=
Time taken to acknowledge a message or poll from the master.

Figure 1. Typical overlapping conversations possible with the Bitbus communications protocol.

Table 2. iRMX 51 Commands

COMMAND	FUNCTION
RQ SEND MESSAGE	Sends a message (a command from the BITBUS master, a response from a slave, or a simple message between tasks on the same BITBUS component) to another task.
RQ WAIT	Waits for an interrupt, an event time-out, a message, or any combination of the three.
RQ CREATE TASK	Causes a new sequence of code to be run as an iRMX 51 task with a specific Function Identification Code.
RQ DELETE TASK	Stops the specified task and removes it from all execution lists.
RQ ALLOCATE	Allocates a fixed-length buffer from the internal 8044 RAM for use as a BITBUS message buffer.
RQ DEALLOCATE	Returns a BITBUS message buffer to the system.
RQ GET FUNCTION ID	Provides a list of the 8 function identification codes representing the tasks currently operating on the microcontroller.
RQ SET INTERVAL	Set the time interval to be used as a separate event-timer for the task.
RQ ENABLE INTERRUPT	Allow external interrupts to signal the microcontroller.
RQ DISABLE INTERRUPT	Stops all external interrupts from signalling the microcontroller.

Bitbus Message Passing. Messages are an integral part of any control-oriented application. Intel's entire Digital Control Module family supports the passing of short control commands, responses to these commands, and status information. A small executive program provides multiple tasking capability so that messages can be managed on a task-by-task basis. All Bitbus components and boards offer the same, simple, message-based interface to user applications that produce and act on control information in small messages. By using standard messages, 8044 tasks can perform I/O operations such as inverting a single I/O bit at a distant Bitbus node, without specially coded communications or bus management software.

BITBUS SPECIFICATIONS

The Bitbus hardware and software interfaces were selected to match distributed control requirements while conforming to established standards supportable with currently available electrical interfaces, cables, and operating systems. However, since some applications require different electrical and mechanical interfaces, the Bitbus connectors and bus interfaces provide additional signals such as power, RTS, DLCK, etc. By using these additional signals, other extensions such as simple fiber-optic communication links and optical isolators can easily be adapted to standard Bitbus connectors.

The Bitbus software interfaces also provide extra customization "hooks." For example, the Function ID Codes and the ability to create new tasks dynamically allow application tasks to take advantage of standard Bitbus services in custom—even proprietary—fashions. To assist in typical and custom designs, a complete Bitbus specification has been published which identifies all facets of Bitbus design.

The Bitbus uses the RS-485 interface as the physical link between controllers. The RS-485 electrical interface is an accepted variation of the common RS-422 interface that allows longer cable segments with more multidrop connections. Repeater stations make it possible to link concentrated control stations with additional remote data collection points. Repeaters are not supported in the Synchronous Mode.

Bitbus Backplane Configurations

Although the Bitbus interconnect is designed as a simple two-wire microcontroller interconnect, it can also be distributed over a backplane. The iRCB 44/10 Remote Controller Board uses a single-wide, 220 mm-deep, Eurocard format and DIN connectors as interface to power, other controllers, and I/O. Table 5 defines the pin-out of the 64-pin DIN connector which facilitates board insertion and maintenance; all I/O and bus connections are on the same connector. The connections are compatible with the standard Intel parallel port adopted by many other vendors. The power pins are compatible with standard Eurocard designations except for the ±12 V provided for the possible use of iSBX modules mounted on Bitbus controller boards.

Since the power and Bitbus connections occupy only a small number (14) of the backplane pins, many custom functions can be implemented on iRCB-type modules. Each module can include a controller and dedicated I/O circuitry designed for a particular application. The iDCM Controller manages all the Bitbus interface and provides the on-chip 8051 controller for local control.

Table 5. Eurocard Connector Pin-Out

DIN PIN #	PIN & SOCKET PIN #	FUNCTION	DIN PIN #	PIN & SOCKET PIN #	FUNCTION
1c		GND	1a		GND
2c		+5V	2a		+5V
3c		DATA*	3a		DATA
4c		DLCK*/RTS*	4a		DLCK/RTS
5c		RGND	5a	1	EXTINT
6c	2	GND	6a	3	PB7
7c	4	GND	7a	5	PB6
8c	6	GND	8a	7	PB5
9c	8	GND	9a	9	PB4
10c	10	GND	10a	11	PB3
11c	12	GND	11a	13	PB2
12c	14	GND	12a	15	PB1
13c	16	GND	13a	17	PB0
14c	18	GND	14a	19	PC3
15c	20	GND	15a	21	PC2
16c	22	GND	16a	23	PC1
17c	24	GND	17a	25	PC0
18c	26	GND	18a	27	PC4
19c	28	GND	19a	29	PC5
20c	30	GND	20a	31	PC6
21c	32	GND	21a	33	PC7
22c	34	GND	22a	35	PA7
23c	36	GND	23a	37	PA6
24c	38	GND	24a	39	PA5
25c	40	GND	25a	41	PA4
26c	42	GND	26a	43	PA3
27c	44	GND	27a	45	PA2
28c	46	GND	28a	47	PA1
29c	48	GND	29a	49	PA0
30c		−12V	30a		+12V
31c		+5V	31a		+5V
32c		GND	32a		GND

INITIAL BITBUS PRODUCTS

The Bitbus microcontroller interconnect is supported by a number of new products. Using a preprogrammed 8044 called the iDCM Controller, the following products provide firmware, additional software, and flexible board-level support for distributed control applications. System-level controllers can take advantage of these modules to extend their I/O into the Bitbus realm. Component-level solutions can make use of the iDCM Controller, or use the software to configure unique solutions that are still compatible with different Bitbus systems.

iRMX 51 Real-Time Executive. The Bitbus firmware is based on a new member of Intel's iRMX Real-Time Operating System family, the iRMX 51 Executive, a very small multitasking executive that supports up to eight user tasks on any of the 8051 family of processors (8051, 8031, 8044, 8744, 8751, etc.). The Executive provides the basic utilities for users to create and maintain tasks, manage interrupts and time intervals, and pass messages between local and remote tasks. Table 2 shows all ten available commands.

The primary operation of the Executive centers around its ability to send messages between tasks residing on the same microcontroller or on another one. As the supporting system for Bitbus firmware, the Executive has been optimized to transfer messages with a minimum of delay. As a general-purpose, real-time executive, it directly supports user tasks located in memory. The first task (task 0) is reserved for a system task, called the RAC function that performs all Bitbus-related functons and provides some user, application-level services.

iRMX 510 DCM Support Package. The package contains software utilities to assist 8044 users in implementing Bitbus-based applications. They include software drivers for interfacing Intel's iRMX 86, 88, 286R, and iPDS ISIS operating systems to Bitbus boards and components. Remote I/O points, from a high-level task, may be controlled using the supplied drivers just as though they were attached directly to the master processor. The 510 package also includes developmental aids, such as a collection of literal definitions and a copy of the Bitbus firmware for use with in-circuit emulators, like the ICE-44, for Intel development systems.

iSBX 344 Bitbus Expansion Module. The module is an 8044-based, double-wide iSBX module, having two 28-pin memory expansion sockets, for driving distributed control systems as either masters or slaves. One socket is equipped with 2 Kbytes of user-accessible memory that is expandable to 8 Kbytes. The other socket may house an additional 64 Kbytes. Users may take advantage of the 8044's features to off-load control and polling functions from the base-board unit.

iRCB 44/10 Bitibus Controller Multimodule. The remote controller board is an 8044-based single-wide Eurocard form-factor board providing the standard Bitbus interface, memory expansion sockets, and clock-support circuitry found on the iSBX 344 Module. It also provides an expansion

Bitbus Robot Example

The Bitbus microcontroller interconnect is well suited for a number of different robot applications. Figure 2 shows a typical robot workstation that includes a machine tool, two robots, a conveyer belt, and a central work-cell controller. The hierarchy of devices shown reflects the recent factory automation trend of connecting more machines to central accounting and control systems. One standard interface (labelled A) at the face-plate of each robot lets users select end-effectors from a variety of sources while maintaining a common control interface to the robot controller. The other standard interface (labelled B) ensures a coordinated work-cell by providing a way to connect robots, conveyers, etc. to the same work-cell controller with a simple and standard control interface.

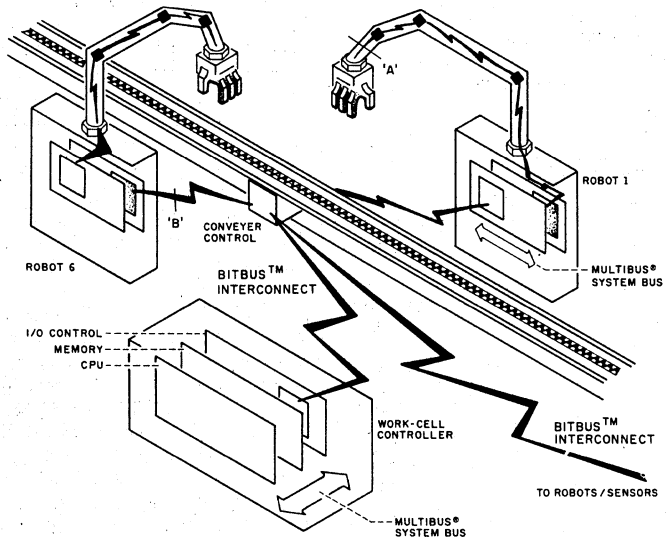


Figure 2. The Bitbus microcontroller interconnect is well suited for a number of different robot applications. Typical robot workstations include a machine tool, two robots, a conveyer belt, and a central work-cell controller. One standard interface (labelled A) at the face-plate of each robot lets users select end-effectors from a variety of sources while maintaining a common control interface to the robot controller. The other standard interface (labelled B) ensures a coordinated work-cell by providing a way to connect robots, conveyers, etc. to the same work-cell controller with a simple and standard control interface.

connector, and 24 lines of bit-programmable I/O. Connection is made by either a standard 10-pin connector or Eurocard 64-pin DIN connector. The iRCB 44/10 form-factor was selected to allow multiple concentrations of controllers.

DESIGNING A FLEXIBLE BITBUS ROBOT

Board and chip level modules can simplify many of the control problems inherent in robot architectures. High-speed central processors are necessary to manipulate positional coordinates and direct individual motor controllers to proper attitudes. Figure 2 shows the Bitbus approach to distributing the control while providing for future expansion and performance enhancements. In this example, the Multibus-based robot controller contains the iSBC 286/10 single board computer, an iSBC 012CX memory expansion board, an iSBC 186/03 single board computer, and two iSBX 344 Bitbus expansion modules. Each board performs a particular system function. The robot drive electronics are housed in a separate Eurocard housing mounted within the robot base. Bitbus connections link the robot controller to the robot, teaching pendant, and work-cell controller. An RS-232 interface is provided to support communication with existing display and control equipment.

Robot Controller. The iAPX 286/287-based computer board provides high-speed computational power to control overall robot motion. Simple high-level commands can be sent over the iSBX connector without regard to their eventual destination because no special software is needed to drive the robot link. The Bitbus controller will automatically send the command to the appropriate control node, ensure proper transmission, and continue polling the slave (interleaved with other commands) until an adequate response is relayed back to the base-board processor. If the computer board uses a multitasking operating system, message traffic can be maintained while the processor is calculating new position information and interacting with the RS-232 link and other processors.

Robot Drive Electronics. A collection of iRCB-style boards in the base of the robot provides servomotor control and sensory feedback to the robot controller. The custom servomotor drive card provides maximum flexibility for each axial motor. Each axial motor control card contains an iDCM Controller that maintains the Bitbus link to the master while performing all the required loop-control calculations. The card also contains inputs for individual limit sensors to maintain desired safety margins. The board is also capable of driving up to 24 auxilliary connections in end-effectors and other peripherals.

The number of degrees of freedom may be increased by adding another axial control card. The Bitbus interface to the robot controller makes it possible to enhance, change, and substitute different robots without having to change hardware or software in the master controller.

CONCLUSION

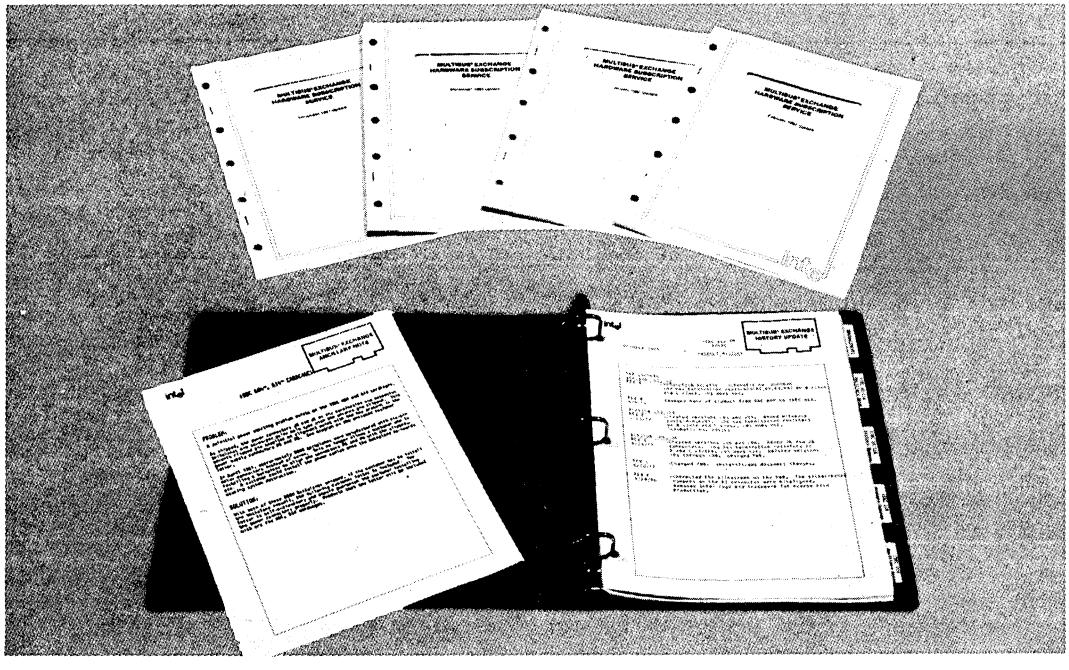
The Bitbus microcontroller interconnect provides a method of connecting controllers in distributed control applications. By means of standard interfaces and specification of new and useful hardware and software interfaces, the Bitbus interconnect can link together many intelligent parts of industrial work-cells, distributed motor and device control, data acquisition, and distributed process control systems. Looking for a standard interconnect, Westinghouse, Unimation, Yasakawa Electric, Mitsubishi Electric, and other well-known industrial control vendors have already turned to the Bitbus as a possible means to remove the final obstacle preventing proper synchronization of high-performance distributed control systems. Using a standard interconnect allows previously isolated controllers to interact with other parts of a distributed control system—even when those parts are from many different vendors.

Miscellaneous Services 15

iMBX 100/110/120/130
MULTIBUS® EXCHANGE
HARDWARE SUBSCRIPTION SERVICE

- **Monthly product updates**
- **History of Engineering Change Orders**
- **Customized to the subscriber's selection of Intel® board level products**
- **Ancillary Notes clarifying and correcting product documentation**
- **Available for iSBC,™ and iSBX™ products**
- **Available for SYP 310**
- **Hardware Reference Manual Revisions for Subscription Products**

The Intel MULTIBUS EXCHANGE hardware subscription service consists of a monthly update publication which provides summary descriptions of Engineering Change Orders that alert and inform subscribers of all the latest developments and/or improvements applicable to their Intel board level products. This timely service allows user to closely track product changes and, by evaluating or implementing these changes, to take advantage of technological and product developments.



October 1981
Order Number: 210276-001

PRODUCTS

The MULTIBUS EXCHANGE hardware subscription service supports Intel manufactured Single Board Computer, Industrial Control Series and MULTIMODULE products. Users can subscribe to any combination of over 60 products.

NEW SUBSCRIBERS

New subscribers receive a history for each subscribed-to product and a handsome notebook in which future updates may be orderly stored.

MONTHLY UPDATES

Each month MULTIBUS EXCHANGE subscribers receive an update containing summaries of Engineering Change Orders and documentation clarification articles relating to each subscribed-to product. Each change description includes the change made, the rationale behind the change, and the user's alternatives concerning updating his product. Subscribers receive the monthly update even if no changes have occured to their products.

In addition, subscribers will receive one new copy of any revised or rewritten subscription product Hardware Reference Manuals in the month immediately following their publication. Hardware Reference Manual revisions incorporate change notices and correct all previously known manual errors.

ORDERS

Orders for the MULTIBUS EXCHANGE are a combination of a base (iMBX 100) order, plus orders for each product to be included (iMBX 110s and/or iMBX 120s). For each MULTIBUS board, power supply, or chassis, an iMBX 110 is ordered. For each MULTIMODULE board, an iMBX 120 is ordered.

CHARGES

For an annual subscription fee, subscribers receive twelve monthly updates. All new subscribers will also receive a history for each subscribed-to product at no extra charge. For additional specified charges, subscribers can purchase the associated documentation and parts kits listed in the MULTIBUS EXCHANGE updates.

ORDERING INFORMATION

Part Number	Description
MBX 100	Base order for the MULTIBUS EXCHANGE hardware subscription service for one year
MBX 110	MULTIBUS EXCHANGE subscription covering one MULTIBUS board, power supply or chassis for one year
MBX 120	MULTIBUS EXCHANGE subscription covering one MULTIMODULE board for one year

INSITE™ USER'S PROGRAM LIBRARY

- **Programs for 8048, 8051, 8080/8085, and 8086/8087/8088 Processors**
- **Accepted Program Submittals Entitle You to a Free Membership or Free Program Package**
- **Worldwide Offices to Serve You**
- **Diskettes, Paper Tapes, and Listings Available for Library Programs**
- **Program Library Catalog Offering Hundreds of Programs**
- **Updates of New Programs Sent During Subscription Period**

Insite, Intel's Software Index and Technology Exchange Library, is a varied collection of programs and routines that have been written by users of Intel microcomputers, single-board computers, and development systems. This expanding library of programs covers a broad range of software tools that includes monitors, conversion routines, peripheral drivers, translators, math packages, and even games. As a library member, you can acquire a copy of any program within the library on any of its available types of media. By taking advantage of the availability of existing library programs, numerous hours of coding and debugging time can be saved and routine or redundant programming operations can be eliminated. The Insite Program Library also serves as a learning tool for individuals unfamiliar with assembly or high-level languages associated with Intel's family of microcomputers.

Membership. Membership in Insite is available on an annual basis. Intel customers may become members through an accepted program contribution or paid membership fee.

Program Submittals. The Insite Library is built on program submittals contributed by users. Customers are encouraged to submit their programs. (Details and forms are available through the Insite Library.) For each accepted program, submittors will receive a choice of up to three free programs (for a maximum value of $300), or free membership with Insite for one year.

Program Library Service. DISKETTES, SOURCE LISTINGS, OR PAPER TAPES are available for every program in Insite. Diskettes are available on single or double density 8″ or iPDS 5¼″. Membership is required to purchase programs.

Insite™ Program Library Catalog. Each member will be sent the Program Library Catalog consisting of an abstract for each program indicating the function of the routine, required hardware and software, and memory requirements.

Insite members will be updated with abstracts of new programs submitted to the Library during the subscription period. For catalog and yearly subscription fee please refer to the Intel OEM Price List or contact the nearest Insite or Intel Sales Office.

INSITE OFFICES ARE WORLDWIDE, WITH FIVE LOCATIONS TO SERVE YOU:

NORTH AMERICA
Intel Corporation
3065 Bowers Avenue
Santa Clara, California 95051
ATTN: Insite User's Program Library
Telephone: 408-987-8080

THE ORIENT
Intel Japan K.K.
5-6 Tohkohdai, Toyosato-cho,
Tsukuba-gun, Ibaraki, 300-26, Japan
ATTN: Insite User's Program Library
Telephone: 029747-8511

EUROPE

Intel Corporation S.A.R.L.
5 Place de la Balance
Silic 223
94528 Rungis Cedex, France
ATTN: Insite User's Program Library
Telephone: 0687-22-21

Intel Semiconductor GmbH
Seidlstrasse 27
8000 Muenchen 2
West Germany
ATTN: Insite User's Program Library
Telephone: 089-5389-1

Intel Corporation (U.K.) Ltd.
Pipers Way
Swindon SN3 LRJ
Wiltshire, England
ATTN: Insite User's Program Library
Telephone: 0793-488-388

SUBMITTAL REQUIREMENTS

Programs submitted for Insite review must follow the guidelines listed below:

Programs must be written in a language capable of compilation and assembly by the currently-supported version of an Intel standard compiler/assembler. Accepted languages are documented in the following manuals available through Intel's Literature Department.

- BASIC-80 Reference Manual, Order No. 980758
- iCIS-COBOL Language Reference Manual, Order No. 980927
- FORTRAN-80 Programming Manual, Order No. 980481
- FORTRAN-86 User's Guide, Order No. 121570
- Pascal-80 User's Guide, Order No. 981015
- Pascal-86 User's Guide, Order No. 121539
- PL/M-80 Programming Manual, Order No. 980268
- PL/M-86 Programming Manual, Order No. 980466
- MCS-48 and UPI-41A Assembly Language Manual, Order No. 980255
- MCS-86 Macro Assembly Language Reference Manual, Order No. 121703
- 8080/8085 Assembly Language Programming Manual, Order No. 980940
- 8086/8087/8088 Macro Assembly Language Reference Manual for 80/85 Based Development System, Order No. 121623
- 8086/8087/8088 Macro Assembly Language Reference Manual for 80/86 Based Development System, Order No. 121703
- 8089 Assembly Language Reference Manual, Order No. 980255
- Microsoft BASIC Compiler Reference Manual, Order No. 121805
- Microsoft BASIC-80 Reference Manual, Order No. 121806
- Microsoft BASIC Reference Book, Order No. 121857
- Microsoft FORTRAN-80 Reference Manual, Order No. 121798
- Microsoft FORTRAN-80 User's Manual, Order No. 121799
- Microsoft M/Sort Reference Manual, Order No. 121809
- Microsoft Utility Software Manual, Order No. 121797

A well-documented source code furnished on an ISIS-formatted 8" diskette, CP/M-formatted 8" diskette, PDS 5 ¼" diskette, or ASCII-coded paper tape.

A source listing of the program must be included. This must be the output listing of a compilation or an assembly. No consideration will be given to incomplete programs or duplications of programs already in the Library.

A link and locate listing.

A demonstration program which assures the validity of the contributed program must be included. This must show the accurate operation of the program.

A complete submittal form.

Licensed software or copyrighted material must be accompanied by a written release from the appropriate, authorized person.

intel®

INSITE™ USER'S PROGRAM LIBRARY SUBMITTAL FORM

Processor

☐ 8048 ☐ 8051 ☐ 8080/8085 ☐ 8086/8087/8088 ☐ Other _____

Indicate the MDS series model the program was created on by checking the appropriate box, and identify other MDS series models the program may be compatible with.

Program Title

Function

Required Hardware

Required Software

Input Parameters

Output Results

Registers Modified:	**Programmer:**
RAM Required:	**Company:**
ROM Required:	**Address:**
Maximum Subroutine Nesting Level:	**City:**
Assembler/Compiler Used:	**State:**
Programming Language:	**Telephone:**

ACKNOWLEDGEMENT AND AGREEMENT

To the best of my knowledge, I have the right to contribute this program material without breaching any obligation concerning nondisclosure of proprietary or confidential information of other persons or organizations. I am contributing this program material on a nonconfidential nonobligatory basis to the Insite User's Library for inclusion in its program library, and I agree that the Library may use, duplicate, modify, publish, and sell the program material without obligation or liability of any kind. The Insite User's Library may publish my name and address, as the contributor, to facilitate user inquiries pertaining to this program material.

Signature _____ Date _____

intel®

INSITE™ USER'S PROGRAM LIBRARY

MEMBERSHIP FORM

I WISH TO BECOME A MEMBER OF INSITE. ENCLOSED IS:

- ☐ CHECK/MONEY ORDER
- ☐ PURCHASE ORDER
- ☐ PROGRAM SUBMITTAL

MEMBER NAME: _____

COMPANY: _____

ADDRESS: _____

_____

_____

TELEPHONE: _____

REFER TO THE INSITE PRICE LIST FOR ANNUAL MEMBERSHIP FEE.

RETURN COMPLETED FORM TO THE NEAREST INSITE OFFICE:

NORTH AMERICA
Intel Corporation
3065 Bowers Avenue
Santa Clara, California 95051
ATTN: Insite User's Program Library
Telephone: 408-987-8080

THE ORIENT
Intel Japan K.K.
5-6 Tohkohdai, Toyosato-cho,
Tsukuba-gun, Ibaraki, 300-26, Japan
ATTN: Insite User's Program Library
Telephone: 029747-8511

EUROPE

Intel Corporation S.A.R.L.
5 Place de la Balance
Silic 223
94528 Rungis Cedex, France
ATTN: Insite User's Program Library
Telephone: 0687-22-21

Intel Semiconductor GmbH
Seidlstrasse 27
8000 Muenchen 2
West Germany
ATTN: Insite User's Program Library
Telephone: 089-5389-1

Intel Corporation (U.K.) Ltd.
Pipers Way
Swindon SN3 LRJ
Wiltshire, England
ATTN: Insite User's Program Library
Telephone: 0793-488-388

iRUG DESCRIPTION

iRUG is the Intel iRMX™ 86 User's Group. It is a non-profit group chartered to establish a forum for users of the iRMX 86 Operating System and to promote and encourage development of iRMX 86 based software.

iRUG membership is free to licensed iRMX 86 Operating System users and to their employees. Benefits of membership include: access to the user's library of iRMX software tools and utilities; membership in local and national chapters; access to the group bulletin board; receipt of quarterly national newsletters; synopsis of software problem reports (SPRs) submitted by members; opportunity to present papers and conduct workshops; invitations to seminars devoted to the use of Intel products.

The user's library, maintained by iRUG, contains software programs written and submitted by members and Intel employees. Programs available range from file or directory manipulation commands and terminal attribute selection utilities to dynamic logon, background job facilities and basic communication utilities.

Programs in the library are available through a telephone dial-up service.

Local and national iRUG chapters provide a forum for members to meet other iRMX Operating System users in an informal setting. At local meetings and the annual international seminar, members can discuss their ideas, share their experiences and techniques, and give feedback to Intel for future improvements and features of the iRMX 86 Operating System. The meetings also showcase new products offered by Intel and other developments in iRMX based software supplied by other companies.

iRUG sponsors a Special Interest Group (SIG) on the CompuServe Information Service. The SIG offers two features, message facilities and an online conference facility. The message facility (buttetin board) allows members to leave and receive messages from other members. These might include problems and solutions regarding the iRMX 86 Operating System or new techniques to be shared. The online conference facility allows users to hold scheduled meetings on any topic. Whatever information a member types at his/her terminal will be displayed at all terminals logged into the conference facility.

"Human Interface" in iRUG's quarterly national newsletter in the United States. It serves as a supplement to chapter meetings by providing: library listings, informtion on the latest releases of products running on the iRMX 86 Operating System; officer messages; member SPRs; release and update plans for the iRMX Operating System; and member articles.

If you are interested in becoming a member of iRUG or desire further information contact the Intel iRUG liason.

Terri Huggett
5200 N.E. Elam Young Parkway
Hillsboro, OR 97123
Mailstop HF2-2-352
(503) 640-7123

UNITED STATES
Intel Corporation
3065 Bowers Avenue
Santa Clara, CA 95051

JAPAN
Intel Japan K.K.
5-6 Tokodai Toyosato-machi
Tsukuba-gun, Ibaraki-ken 300-26
Japan

FRANCE
Intel
5 Place de la Balance
Silic 223
94528 Rungis Cedex
France

UNITED KINGDOM
Intel
Piper's Way
Swindon
Wiltshire, England SN3 1RJ

WEST GERMANY
Intel
Seidstrasse 27
D-8000 Munchen 2
West Germany

Printed in U.S.A./C-793/1284/100K/RRD MB
OEM Systems

ISBN 0-917017-17-X