THE CP/M USER'S JOURNAL $2.95

Microsystems

November/December 1982

A hard look at the merits and demerits of CP/M-86 and MS-DOS


SIERRA DATA SCIENCES
GREAT IDEAS...
Down to Earth
Products
(AVAILABLE SOON)
SDS-S100 CHASSIS
6 Slot Motherboard
Complete power supply
Utilize regular or Thinline drive
SDS-MULTIPLEXER/DISPLAY
Three RS232C 1 to 2 Switches
Two Seven Segment Status Display
(Can be used as a line monitor for data
communications link)
(route RS232 to one of two devices)
NEC-FLOPPY DISK DRIVE
Double Sided
Single Density/Double Density
Up to 2.4 Megabyte
SPECIAL PRICING
SDS-HARD DISK INTERFACE
Micropolis 1220 Series
Interface Adapter
SDS-SLAVE
SDS-ZSIO
SDS-MASTER
SDS-SINGLE BOARD COMPUTERS
Z80A CPU 64K Bank Switch Memory
2 RS232 Channels 4 Timers IEEE 696 Buss
Interface
4 Parallel ports
NEC 765 FDC with PLL to all Shugart
compatable drives (SDS-Master only)
HARDWARE
SOFTWARE
SIERRA DATA SCIENCES
Fresno CA / Marketing Division, 21162 Lorain Ave., Fairview Park, Ohio 44126
(216) 331-8500 Telex. 980133 WDMR
CIRCLE 104 ON READER SERVICE CARD

Microsystems

Staff

Sol Libes . . . . editor
Chris Terry . . . . technical editor
Jon Bondy
Anthony Skjellum
. . . . .contributing editors
Don Libes
Fred Gohlke
David Gewirtz
Randy Reitz. . . . . . . assisting editors
Ann Ovodow
. . . . editorial coordinator
Nancy Metz . . . . . editorial secretary

Mariano Nicieza . . . . art editor

William L. Baumann
. . . . .operations manager
Patricia Kennelly. . . . . . comptroller
Ethel Fisher. . . . . bookkeeping
Mary McNeice
Frances Miskovich. . . . . . secretaries
Carol Vita . . . . fulfillment
Sue DeMark . . . . . customer service
Jennifer Burr. . . . . retail sales
Ron Antonaccio
. . . . shipping & receiving

Jeff Weiner . . . . advertising director
Karen Jacobs
. . . . advertising coordinator

MICROSYSTEMS (ISSN #0199-7955) is published bimonthly by Ahl Computing, Inc., a subsidiary of Ziff-Davis Publishing Company, One Park Avenue, New York, N.Y. 10016. David Ahl, President; Elizabeth B. Staples, Vice-President; Selwyn Taubman, Treasurer; Bertram A. Abrams, Secretary.

Second Class postage paid at New York, NY 10016 and at additional mailing offices. POSTMASTER: Send address changes to MICROSYSTEMS, PO Box 1987, Morristown, NJ 07960. Subscriptions are $24.97 for 12 issues. Canadian prices are $5.00 per year additional; other foreign $8.00 per year additional (U.S. currency only). For information or questions about subscriptions phone: (800) 631-8112.

Editorial correspondence is welcomed and should be sent to: MICROSYSTEMS, One Park Avenue, New York, NY 10016. Phone: (212) 725-6856.

For information on commercial advertising, write to: MICROSYSTEMS Advertising Dep't., One Park Avenue, New York, NY 10016, or call Jeff Weiner at (212) 725-7957.

MEMBER MITA

NOW: A COMPLETE CP/M PASCAL FOR ONLY

$29.95!

HERE'S THE PASCAL YOU'VE BEEN READING ABOUT. AND, WITHOUT EXCEPTION, THE REVIEWS CALL JRT PASCAL A SUPER PRODUCT FOR AN AMAZING PRICE!

InfoWorld
Software Report Card

JRT Pascal Version 2.0

	Poor	*Fair*	*Good*	*Excellent*
Performance	☐	☐	☐	☑
Documentation	☐	☐	☑	☐
Ease of Use	☐	☐	☐	☑
Error Handling	☐	☐	☐	☑

From INFOWORLD magazine, August 16, 1982

Goodbye BASIC, PL/1, COBOL—hello PASCAL! Now, to make this most advanced language available to more micro users, we've cut our price—to an amazing **$29.95!** This astonishing price includes the complete JRT Pascal system on diskette and the new, comprehensive user manual. Not a subset, it's *a complete Pascal for CP/M.** Check the features below.

THIS IS THE SAME SYSTEM WE SOLD FOR $295!

So how can we make this offer?—why the unbelievable deal? Very simply, we think all software is overpriced. We want to build volume with the booming CP/M market, and our overhead is low, so we're passing the savings on to you.

AND AT NO RISK!

When you receive JRT Pascal, look it over, check it out. We invite you to compare it with other systems costing ten times as much. If you're not completely satisfied, return the system—with the sealed diskette unopened—within 30 days and your money will be refunded in full! *THAT'S RIGHT–COMPLETE SATISFACTION GUARANTEED OR YOUR MONEY BACK!*

In addition, if you want to copy the diskette or manual—so long as it's not for resale—it's o.k. with us. Pass it on to your friends! BUT ACT TODAY—DON'T DELAY ENJOYING PASCAL'S ADVANTAGES—AT $29.95, THERE'S NO REASON TO WAIT!

New 125-page user manual and 5¼″ or 8″ diskette

True dynamic storage

Advanced assembler interface

Fast one-step compiler; no link needed

Efficient compiler needs only 85K diskette space

Maximum program size more than 200,000 lines

Random files to 8 megabytes with variable length records

64K dynamic strings

Activity analyzer prints program use histogram

Extended CASE statement

Graphing procedures

Statistic procedures

14 digit BCD FLOATING POINT arithmetic

More than 200 verbal error messages

Separate compilation of auto-loading external procedures

No limits on procedure size, nesting, or recursion

JRT PASCAL

Send to

JRT SYSTEMS
1891—23rd Avenue
San Francisco, CA 94122

Phone

415/566-5100

O.K. You've sold me. Send me JRT Pascal; I understand that if I'm not completely satisfied, I can return it within 30 days—with the sealed diskette unopened—for a full refund.
(Please allow 2-3 weeks for delivery.)

I need the **5¼″** diskette for ☐ Northstar ☐ Osborne ☐ Apple-CP/M ☐ Heath, Hard Sector ☐ Heath, Soft Sector ☐ Superbrain.
I need the ☐ **8″** SSSD diskette.

Name ____________ Address ____________

City ____________

State ____________ Zip ____________

☐ Check ☐ C.O.D. ☐ MasterCard ☐ VISA
(CA residents add sales tax. Add $6 for shipping outside North America.)

Card # ____________ Exp. ____________

Signature ____________

*CP/M is a Digital Reserach TM. A 56K CP/M system is required.

CIRCLE 114 ON READER SERVICE CARD

Teletek's SBC~I Just Met Its Master

SYSTEMASTER® masters the S-100 bus and SBC-1 slaves!

Teletek's SYSTEMASTER® single board computer is not only a stand-alone computer, but is also the master controller for running multiuser/multiprocessor systems.

SYSTEMASTER® becomes the heart of a powerful and fast network of individual user processors when used with Turbodos or Micro Mikes operating systems — up to 16 users. RS-422 capability is available when parallel ports are connected to Teletek's PSC board.

Teletek's SBC-1, a board designed for use in high performance multiprocessing systems, features 64k or 128k of on board RAM, 64k RAM Cache Buffer for each user with Turbodos, and 2k FIFO. SBC-1 is the logical choice for use with SYSTEMASTER®.

SBC~I and SYSTEMASTER®

A powerful combination from the S-100 manufacturer that always brings you the best.

TELETEK

9767 Business Park Drive
Sacramento, CA 95827 (916) 361-1777
Telex 4991834 TELETEK
Dealer inquiries invited.

CIRCLE 38 ON READER SERVICE CARD

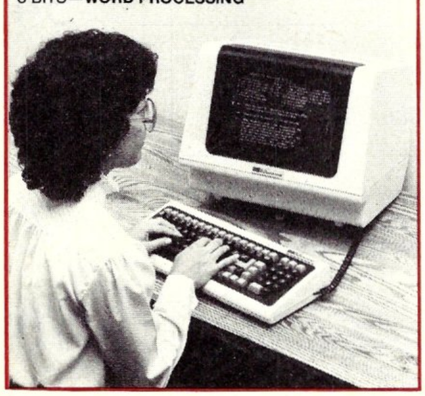
Contents

Microsystems

Volume 3 / Number 6
November / December 1982

Finally, a MULTI-USER micro for professionals by G&G Engineering

8 BITS—**WORD PROCESSING**

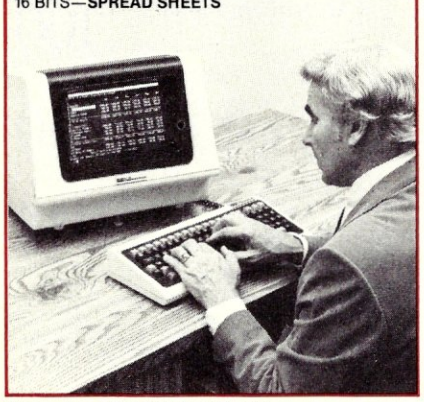


Shown: WordStar™ from MicroPro

EXPANDABLE

You can start with a single user installation and add users as your needs grow. The G&G system can be easily expanded in field—just add more memory.

16 BITS—**SPREAD SHEETS**



Shown: SuperCalc 86™ from Sorcim

Hard Disk Based Multi-User Systems start at $10,500

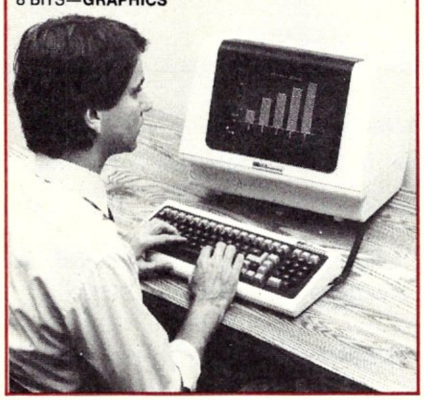
MP/M 8-16 is a proprietary implementation of Digital Research's MP/M 86 operating system, configured for CompuPro by G&G Engineering. CompuPlot is a trademark of G&G Engineering. CP/M and MP/M are registered trademarks of Digital Research. SuperCalc and SuperCalc 86 are trademarks of Sorcim. dBASE II is a trademark of Ashton-Tate. WordStar is a trademark of MicroPro. CompuPro is a trademark of Godbout Electronics.

MULTI-USER MP/M™ 8-16™

Up to seven users can run their own programs with their own protected data or share common programs and data. High speed DMA hard disk and cache memory give this system fast access to data, so you avoid bottlenecks.

G&G's MP/M 8-16 systems have been shipping since March, 1982. They are field proven!

8 BITS—**GRAPHICS**



Shown: CompuPlot™ from G&G Engineering

8 and 16 BIT CP/M™ COMPATIBLE

Each user can run the vast inventory of 8 bit CP/M programs, or choose from the library of powerful new 16 bit CP/M 86 software. Your past investment in CP/M software is protected, while at the same time the door to the 16 bit world is opened to you.

SUPPORT

We wrote the book on 8/16 bit multi-user systems. Our phone support is the best in the business. Plus, CompuPro components in G&G systems carry a two year factory-backed warranty.

CIRCLE 11 ON READER SERVICE CARD

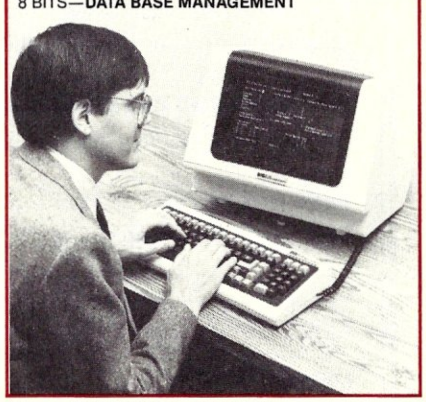
16 BITS—**ACCOUNTING**

Shown: GL by Structured Systems Group

UPGRADABLE

The G&G system is based on the IEEE 696/S-100 bus and uses CompuPro's state-of-the-art components. Our 20 slot bus guarantees that you will never be left in a corner with obsolete technology. Your investment in hardware is protected.

8 BITS—**DATA BASE MANAGEMENT**



Shown: dBASE II™ from Ashton-Tate

G&G Engineering is an authorized CompuPro systems center.

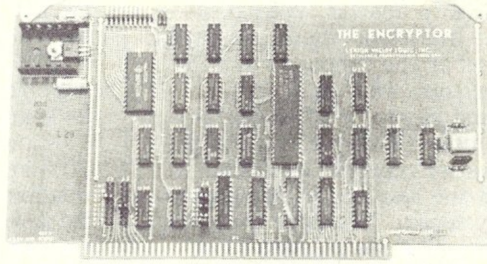
G&G ENGINEERING

1922 Republic Avenue, San Leandro, California 94577 (415) 895-0798
230 California Street, Suite 207, San Francisco, California 94104

The Encryptor™

Models: IIX-100A IIX-100B

DATA SECURITY PLUG-IN PERIPHERAL FOR S-100 / IEEE-696 COMPUTERS.



Applications:

- ☐ Telephone line security.
- ☐ Payroll security.
- ☐ Sensitive information protection.
- ☐ Security for data stored in diskettes, hard disks, tapes, and I.C. memories.

Used by:

- ☐ Accountants.
- ☐ Attorneys.
- ☐ Banks.
- ☐ Businesses.
- ☐ Consultants.
- ☐ Doctors.
- ☐ Educators.
- ☐ Law enforcement agencies.
- ☐ State and local governments.
- ☐ Stockbrokers.

Features:

- ☐ A high degree of security at an affordable price.
- ☐ Implements the National Bureau of Standards Data Encryption Standard (D.E.S.).
- ☐ High speed data ciphering processor.
- ☐ IIX-100A: 1.3 megabytes per second.
- ☐ IIX-100B: 1.7 megabytes per second.
- ☐ Encryption and decryption of data.
- ☐ 3 ciphering modes.
- ☐ Over 72 quadrillion 56 bit keys possible.
- ☐ MIL-STD-883, Class B High reliability I.C.'s.
- ☐ Plug selectable I/O port addresses and interrupts.
- ☐ Gold plated fingers, card ejectors and solder mask.
- ☐ Assembled and tested.
- ☐ 1 year limited factory warranty.

Lehigh Valley Logic

Lehigh Valley Logic, Inc.
2503 North Court
P.O. Box 3216
Bethlehem, Pennsylvania 18017
(215) 865-1222

CIRCLE 74 ON READER SERVICE CARD

Editor's Page

by Sol Libes

A report to the reader

In the January/February issue of *Microsystems* we included a two-page questionnaire, containing 32 questions, for our readers to answer. We asked many questions about our readers, what they thought of *Microsystems* and what they wanted to see in *Microsystems*. We took the first 250 questionnaires we received and tabulated the results for this preliminary report. We hope that the information we gleaned from this effort will help us tailor *Microsystems* more closely to the needs of our readership.

It is a very large and tedious task to analyze the data and put it into report form. The full report is still in preparation. In the meantime, however, here is some preliminary data.

The average *Microsystems* reader

First of all, we are most impressed with our readers. Almost 40% have Master's or Ph.D. degrees, more than 80% have at least one college degree, and 97% have some college work. 99% of our readers are male and 98% are 25 or more years old (average age—38). 55% belong to a scientific, engineering or computer professional society. In other words, *Microsystems* readers are mature adults in the prime of their working life; they are serious users of computers.

The average reader earns $38,000 a year, and one-third of our readers earn $50,000 or more. The median amount of money that a reader has spent on his microcomputer system is $5,800; 30% have spent over $8,000, and 18% over $10,000. The average reader plans to spend another $2,000 or more on his system this year, for both hardware and software.

98% of our readers are professionals who also use computers at work. The average reader has access to three microcomputer systems. 70% use them in professional applications, 33% use them in research and development, 21% use them in education, and 83% also have them for personal use. 76% said they either make or strongly influence computer purchasing decisions in their companies.

37% of the readers are engineers/scientists/technicians, 24% are programmers/analysts, 13% own their own businesses, 10% are professionals (law, medicine, accounting, etc.), and 5.5% are educators. Fewer than 1% are students.

The typical *Microsystems* reader is thus well educated, mature, affluent, a professional, and uses microcomputer systems at work as well as home.

What systems do readers own?

We asked readers to identify which S-100 system they owned. It is interesting to note that the average *Microsystems* reader owns more than one S-100 system. In fact the average was 2.2 S-100 systems. Furthermore, they often own systems from different manufacturers. Therefore the following percentages, based on the number of respondents rather than systems, total more than 100%.

Almost one-third said they had Godbout-based systems. More than 25% have "homebrew" systems, which probably means that the system is composed of components from several different manufacturers rather than built from scratch. Almost 22% own North Star systems and more than 20% own Morrow Designs systems. More than 20% have SSM systems, and almost 20% have Cromemco systems. More than 18% have IMSAI systems, and more than 15% have CCS systems. Vectorgraphic was represented with over 13%.

Although we did not ask, many readers indicated that they also owned systems other than S-100s. Here, the leaders were Radio Shack TRS-80, Intel Multibus, and Apple. When these are added to the total of computers owned by readers, it means that the average reader owns 2.74 systems.

The average reader has spent about $5,000 on his system.

Reader preferences

We asked readers for their opinions about the content and direction of *Microsystems*. We want *Microsystems* to meet our readers' needs as much as possible. We therefore asked several questions that we hoped would indicate whether we were on the right track and what we should be emphasizing in the future.

We asked readers to indicate what they thought were the four best articles that appeared in 1981. Approximately 200 people answered this question. The following table shows the top 15 articles, the authors and the number of readers choosing them:

1. CP/M Connection	*C. Terry*	*50*
2. Introduction to CP/M	*J. Epstein*	*37*
3. Intro to the C Language	*D. Gewirtz*	*29*
4. S-100 Standard Update	*S. Libes*	*23*
5. S-100 Clock/Calendar	*F. Deadrick*	*20*
6. CP/M Bus	*A. Skjellum*	*19*
7. Little Ada	*R. Kenyon*	*18*
8. 16-Bit Operating Systems	*S. Libes*	*15*
9. Double-Density Disk Review	*B. Weidemann*	*15*
10. Godbout Dual CPU Review	*B. Ratoff*	*14*
11. Intro to Data Communications	*L. Hughes*	*13*
12. Three S-100 Z80 CPUs	*W. Machrone*	*13*
13. Three S-100 Modems	*M. Zeiger*	*13*
14. Communications Explosion	*F. Lepow*	*10*
15. S-100 CPU Cards	*S. Libes*	*10*

When readers were asked to rate the regular departments in each issue as to which was most valuable, the following rank order developed:

sweet sixteen...

Come of Age Into the World of **16-BIT** Computing & Networking...

without the growing pains*

*Without sacrificing your 8-bit investment in Hardware and Software

Our DISCOVERY Multiprocessor is the **first and only** multi-user multiprocessing microcomputer system that allows for the simultaneous operation of **both** 8-bit , Z-80 based and **16-bit, 8086/8087** based processors within the same system. Furthermore, it is now possible to combine individual DISCOVERY Multiprocessor stations into **Local Area Networks** with capability surpassing most minicomputers and mainframes.

THE END USER is guaranteed access to the World's most abundant supply of computer languages and software. In DISCOVERY, the industry standard **CP/M-80 and CP/M-86**** run simultaneously and concurrently while maintaining compatibility between 8-bit and 16-bit databases. With DISCOVERY, the new generation of software designed specifically for multi-user applications is available for your use... **Right Now!**

As **A SYSTEM HOUSE**, you will find that DISCOVERY easily brings all your existing 8-bit development tools and high-level applications software to the strength of the 16-bit World. Just think, access to a million bytes of addressable memory and lightning fast execution speeds. With the powerful file and record locking facilities of our own **dpc/os**©** (Distributed Processing Computer Operating System), you can create flexible solutions to meet your customers' needs. Let your sales grow into the ever expanding multi-user market.

DEALERS and DISTRIBUTORS, you will find it easy to sell our DISCOVERY system to your customers with the assurance that any time they chose, they can grow painlessly into the **Sixteen-Bit** World and into **Networking.** The DISCOVERY is the only system in the microcomputer marketplace that will provide your customers with the natural growth path...from 8-bit single user computers to full-blown multi-user networks of **16**-bit work stations and with up to **SIXTEEN** users in each complex!

**CP/M-80 and CP/M-86 are TMs of DIGITAL RESEARCH INC.
dpc/os© is a TM of ACTION COMPUTER ENTERPRISE, INC.

Take ACTION! Call us NOW... at (213) 793-2440

Systems complete with Hard Disks available for immediate delivery. Dealer, Distributor & OEM inquiries are invited.

ace

Action Computer Enterprise, Inc. ○ 55 West Del Mar Blvd. ○ Pasadena, CA 91105 USA
TWX-910-588-1201 ACTION PSD ○ Telephone (213) 793-2440
On the East Coast: Please Contact MicroSystems International ○ (617) 655-9595

CIRCLE 78 ON READER SERVICE CARD

Editor's Page
continued . . .

1. *CP/M Bus*
2. *News & Views*
3. *New Products*
4. *Editor's Page*
5. *Software Directory*
6. *Letters*
7. *North Star Topics*

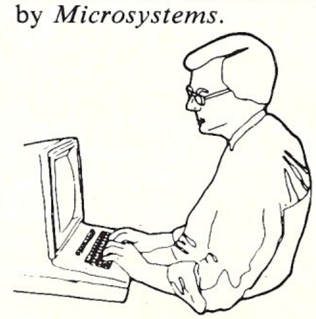
Next, we asked readers to indicate the types of articles they prefer. The following shows the rank order:

1. *CP/M tutorials*
2. *Software reviews*
3. *Software applications*
4. *System software*
5. *Programming tutorials*
6. *Assembler programs*
7. *Hardware reviews*
8. *Hardware tutorials*
9. *Basic programs*
10. *Systems reviews*
11. *Pascal programs*
12. *C programs*
13. *PL/I programs*

Asked to specify the topics they were most interested in, readers responded as follows:

1. *CP/M*
2. *Graphics*
3. *S-100*
4. *Database management*
5. *North Star systems*
6. *MP/M*
7. *C Language*
8. *Hardware reviews*
9. *Multiuser systems*
10. *Utility programs*
11. *MS-DOS*
12. *basic programs*
13. *16-bit systems*
14. *Data communications*

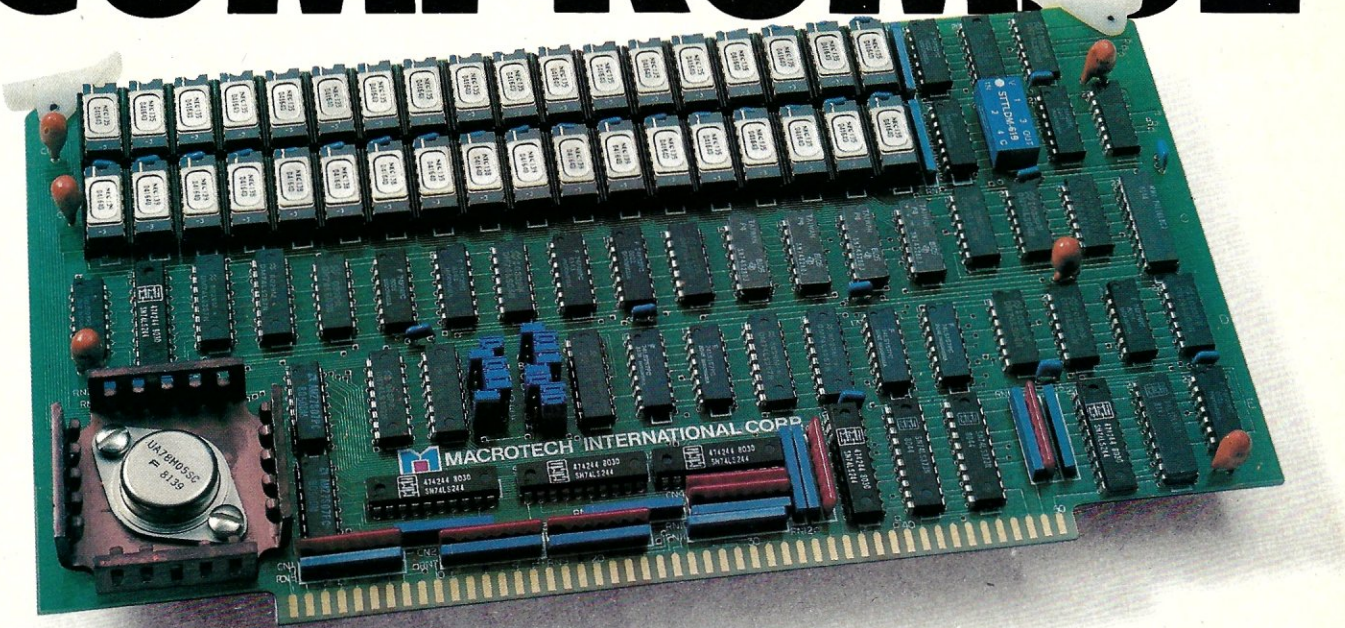
We wish to thank all the readers who took the time and effort to complete the questionnaire and mail it to us. We feel that this was a very worthwhile experience which will be reflected in the direction taken by *Microsystems*.



Sol Libes

THE MULTI-USER FRIENDLY 256K MEMORY BOARD WITH

NO COMPROMISE



No Compromise in Performance

- ☐ Main-frame type memory mapping for optimal BANK SWITCHED applications or 24-BIT extended addressing
- ☐ Operates in *any* S-100 System IEEE/696† or non IEEE
- ☐ DMA operation fully supported†
- ☐ 8/16 data transfer protocol†
- ☐ Up to 6 MHZ with no wait states; Up to 10 MHZ with automatic wait state assertion
- ☐ Parity error detection†
- ☐ Double parity in 16-BIT operation

No Compromise in Features

- ☐ All memory/refresh cycle timing is generated ON BOARD providing easy CPU independent characterization
- ☐ Ideal for use with the new generation of operating systems: CP/M-80,™* CP/M-86,™* MP/M-11,™* MP/M-86,™* OASIS,™** UNIX™***

No Compromise in Quality

- ☐ Multilayer board and bus signal filtering for noise-free operation
- ☐ Thorough 100% final test and burn-in

No Compromise in Versatility

- ☐ Manual provides source listing and complete installation guide for MP/M-II™* BANK SWITCHED multi-user applications
- ☐ For the CP/M-2.2™* user: Manual includes implementations of 'Virtual Disk' for solid state disk/file applications

No Compromise in Customer Support

- ☐ Comprehensive technical manual
- ☐ User hot-line
- ☐ Full one year parts and labor warranty

†In accordance with IEEE-696 standard
Registered trademarks of: *Digital Research, **Phase I, and ***Bell Labs.

ORDER NOW

(213) 887-5737

If you wish, send a check or money order for $1,379 (in California add 6% sales tax where applicable).

Manual available at $25.00 each refundable with order.

MACROTECH International Corp., 22133 Cohasset St., Canoga Park, California 91303

MACROTECH International Corp.,
22133 Cohasset St., Canoga Park,
California 91303

CIRCLE 28 ON READER SERVICE CARD

News & Views

by Sol Libes

Rumor

Motorola is rumored testing out 16MHz versions of its 68000 microprocessor . . . .

New show to feature CP/M

Northeast Expositions Inc. will produce "CP/M '83", an international conference and exposition for developers, distributors, retailers, and end users of CP/M software. The show will be held Thursday thru Sunday, January 20–23, at San Francisco's Moscone Center. The event is sponsored by Digital Research.

There will be many CP/M-related workshops. Noted leaders from the software industry will conduct sessions exploring CP/M applications, technical information, development aids, and support services.

For more information write Northeast Expositions, 824 Boylston St., Suite 202, Chestnut Hill, MA 02167 or call (617) 739-2000.

CP/M68K report

CP/M68K for Motorola 68000-based machines is now in final testing and is expected shortly. Originally written for a machine that will soon be introduced by Hitachi, it is rumored that Motorola will also be distributing it.

CP/M's great popularity is due in great measure to the large number of existing programs, both public domain and commercial, which run under it. However, these programs are written in 8080/Z80 code and will not run on 68000-based machines. The question now is: When will someone introduce a translator program that will automatically convert all the CP/M-80 goodies to CP/M-68K?

New Microsoft software

Microsoft is expected to release version 2 of MS-DOS shortly, enhanced by multitasking capability. MS is also rumored working on a version that will run as a task under multiuser Xenix, so that users can run MS-DOS software on Xenix systems. One last rumor is that Microsoft may soon have a word processor, possibly called "MultiWord," to go with their "MultiPlan"—a planning tool for economic forecasts, etc.

432 news

Some benchmark tests that included the new Intel 432 32-bit micro were recently run at Berkeley (*Computer Architecture News*, June 1982). The benchmarks included four programs (string search, prime number sieve, bin-packing puzzle, and Ackermann with 170,000 procedure calls) written in Ada, C, and Pascal. The 432 was an Intel 4MHz release 2 system with programs written in Ada. The 8086 system ran at 5MHz with a Pascal compiler. The 68000 systems were a Dual Systems 8MHz S-100 with MIT C compiler, an Exormacs 8MHz with Motorola Pascal compiler, and a Motorola 16MHz system that had no wait states. The VAX system had the VMS Pascal, Unix C, and the Berkeley Unix Pascal Compilers.

The timings measured were: 432 = 0.05 VAX, 8086 = 0.4 VAX, 8MHz 68000 = 0.6 VAX, and 16MHz 68000 = 1.75 VAX.

Those interested in obtaining data on the 432 should know that documents in the following list are available from Intel. Intel will furnish this material, subject to availability, at no charge for University use (request must be on University letterhead and indicate intended use and quantity desired). Request it from: Intel, OMSD University Program, HF2-2-340, 5200 NE Elam Young Pkwy, Hillsboro OR 97123.

- Intellec 432/100 Evaluation & Education System Data Sheet
- Getting Started on the Intellec 432/100
- Object Programming Language User's Guide
- Object Builder User's Guide
- iSBC 432/100 Processor Board Hardware Reference Manual
- iAPX 43201/43202 VLSI General Data Processor Data Sheet
- iAPX 43203 VLSI Interface Processor Data Sheet
- iAPX 432 Cross Development System Data Sheet
- System 432/600 32-bit Extensible Computer System Data Sheet
- Introduction to iAPX 432 Architecture
- iAPX 432 Object Primer
- iAPX 432 General Data Processor Architecture Reference Manual
- iAPX 432 Interface Processor Architecture Reference Manual
- Reference Manual for the Ada Programming Language
- Reference Manual for the Intel 432 Extensions to Ada
- iMAX 432 Operating System Reference Manual
- Introduction to the Intel 432 Cross Development system
- Intel 432 Cross Development System VAX/VMS Host User's Guide
- Asychronous Communication Link User's Guide
- Intel 432 Cross Development System Workstation User's Guide
- System 432/600 System Reference Manual
- System 432/600 Diagnostic Software User's Guide

Z80 strong as ever

ZILOG reports that sales of Z80 chips are stronger than ever (probably because of the ZX81) and that within the last three months they closed two deals for one million each, and three for one-half million each. Z80 prices in this quantity are

News & Views
continued . . .

typically around $3 apiece. Zilog reports that it currently is supplying 60% of the Z80 market, Mostek is supplying 20%, and SGS and Sharp are supplying the remaining 20%. Zilog expects to ship about six million Z80 chips this year, up from four million last year. This means that about 10 million Z80s will be sold this year.

Dataquest, a marketing research outfit, recently reported that 50% of the new microcomputer products introduced at the recent NCC contained Z80s.

Current Zilog Z80s are rated for 6 and 8MHz operation. Next year Zilog expects to introduce a CMOS version of the Z80.

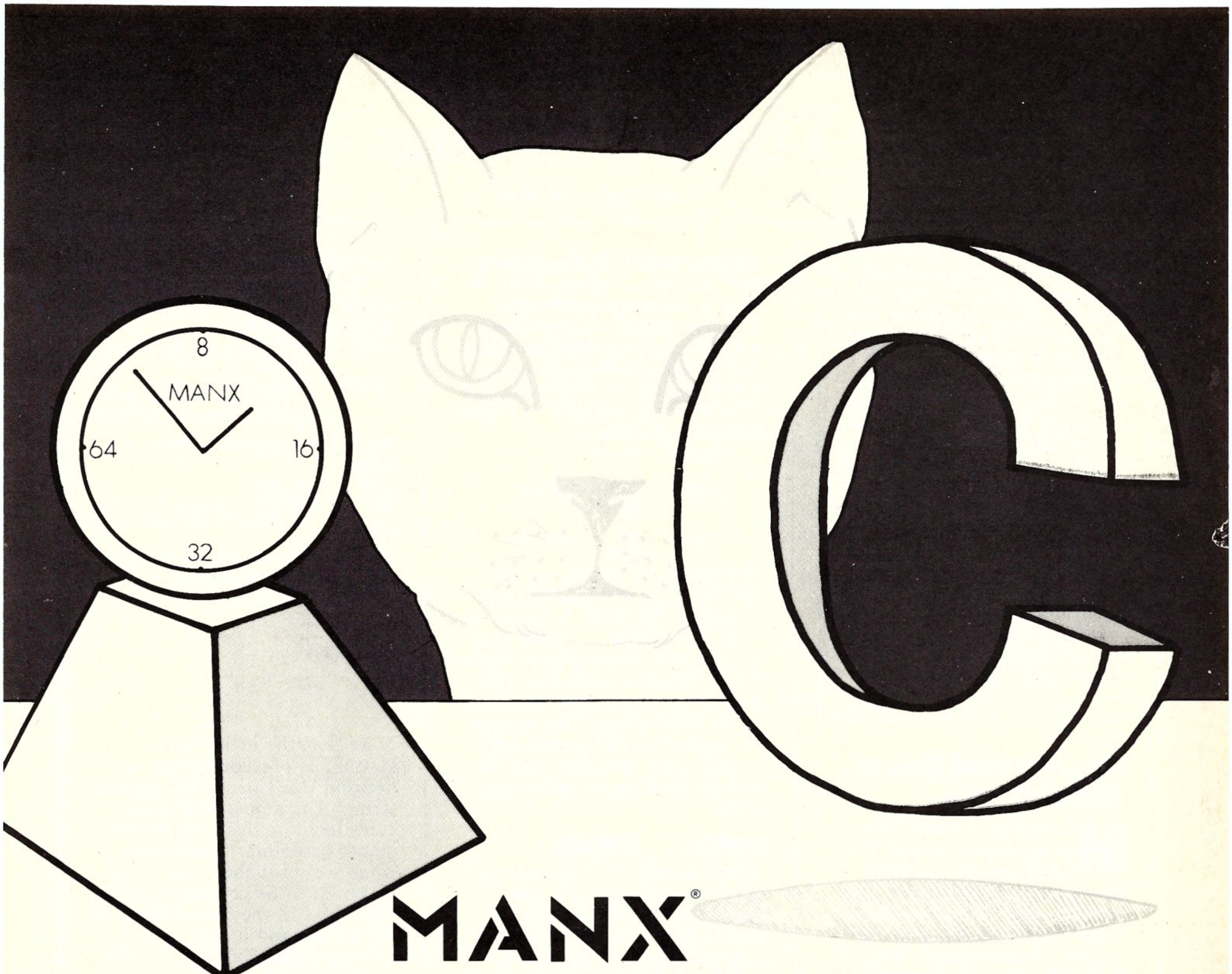
User group news

USUS (UCSD Pascal Users' Society) has formed DEC-SIG Special Interest Group for users of the UCSD P-System on DEC computers. Membership in DEC-SIG is free to USUS members. USUS membership is $20/yr. For information contact: Eli Willner, DEC-SIG chairman, Datronics, Inc., 675 Third Ave, NY NY 10017; Compuserve 70210,317 or Telemail EWillner/US.US.

There is a new North Star users group that already has over 700 members. It is called INSUA. It publishes a newsletter and has a software library. Write: INSUA, Box 2789, Fairfield CA 94533.

News bits

The EPROM may soon be a thing of the past as Intel, SEEQ, General Instrument, National Semiconductor and Hitachi are expected shortly to introduce 16K, 5V EEPROMs (electrically erasable). . . . Lobo Drives, Goleta CA, has introduces an $800 Z80-based system that runs CP/M or LDOS (TRS-80 compatible DOS). The unit includes keyboard, 64K RAM, clock/calendar with battery backup and user-definable character set. A CRT monitor is $150. No word on disk system cost.



MANX®

A NEW DIMENSION IN SOFTWARE

For Quality, Performance, Price, and Features *There Is No Comparison*

C COMPILERS - COMMON FEATURES:

• UNIX VER 7 compatibility (Full implementation of "C" as defined in "The C Programming Language" by Brian W. Kernighan and Dennis M. Ritchie.") • standard float, double, and long support • run time library with full I/O and source • fast compilation and execution • ALL C Operators • supports assembler subroutines • supports modular programming with true extern support • extern, static, auto, and register storage classes • unions, structures, multi-dimension arrays • short and unsigned data types • if, while, do, for, switch, case, goto • #define, #include • #ifdef, #ifndef, #else, #endif • command line arguments using argc and argv • DATA FORMATS: char- 8 bits • short- 16 bits • long- 32 bits • float- 32 bits • double- 64 bits.

AZTEC C II CP/M (MP/M) **$199**

NEW!!! Overlays • Library utility • produces relocatable 8080 source code (optional Z80 instructions) • relocating assembler and linker supplied • MICROSOFT M80 interface • SID/ZSID interface • requires 48K TPA (Apple requires 16K card).
(Also available for HEATH HDOS).

AZTEC C II APPLE DOS **$199**

• relocating assembler and linker supplies • APPLE SHELL • VED editor • library and other utilities • requires 16K card.

C86 IBM PC • MSDOS • CP/M-86 **$249**

• directly produces 8088/8086 code • linker supplied

K & R BOOK . . $18 Manuals $30

ORDER BY PHONE OR MAIL
Specify products and disk format

CP/M FORMATS: 8"STD, HEATH, APPLE, OSBORNE, NORTHSTAR,...

OUTSIDE U.S.A. - Add $10 In N.J. add 5% sales tax

MANX®
software systems
Box 55, Shrewsbury, N.J. 07701
(201) 780-4004

CIRCLE 29 ON READER SERVICE CARD

The CP/M Bus

by Anthony Skjellum

This column is not only an ongoing tutorial on various topics related to CP/M, but is also intended as a forum in which readers may ask questions or bring up problems they have encountered in using CP/M. Your questions are welcomed—we can't promise a reply in the very next issue, but they *will* be answered. Meanwhile, until we get some questions from you, Anthony Skjellum will discuss various aspects of assembly language macro instructions provided by macro assemblers such as Digital Research's MAC and Microsoft's M-80.

—Editor

Macros and Macro Assemblers

Macro instructions allow the programmer to produce repetitive code and/or data sequences concisely, and possibly with variations. Simple macros produce the same sequence of instructions or data each time they are invoked. More complicated macros may produce code that depends on one or more parameters specified when the macro is invoked.

Macro instructions come in two forms: in-line and stored. In-line macros are macro instructions that are built into the assembler. These include the REPT . . . ENDM, IRPC . . . ENDM, and IRP . . . ENDM groups. In-line macros will be considered first.

REPT . . . ENDM:

This instruction allows a sequence of instructions to be produced repeatedly. It has the form:

```
lbl1:   REPT    expr
        stmt#1
        stmt#2
        ...
        stmt#N
lbl2:   ENDM
```

where the labels lbl1, lbl2 are optional assembly language labels, stmt#1 . . . stmt#N are assembly language statements. The statements are repeated until the expression **expr** evaluates to zero. Here is an example of REPT . . . ENDM in use. This example produces a 7-byte sequence of zeroes.

```
VALUE   SET     7
        ;
AREA:   REPT    VALUE
        DB      Ø
        ;
VALUE   SET     VALUE-1
        ENDM
```

The data area is labelled by AREA:. The symbol VALUE is used as an index in the iteration. The REPT . . . ENDM sequence repeats as long as VALUE is nonzero. Note that the SET pseudo-operation is similar to EQU except that it may be applied more than once to an assembly language symbol without causing an assembler error.

IRPC . . . ENDM:

IRPC stands for "indefinite repeat—characters." As the name implies, the purpose of this in-line macro is to handle character data. This macro group takes the form:

```
lbl1:   IRPC    IDENT,char_list
        stmt#1
        stmt#2
        ...
        stmt#N
lbl2:   ENDM
```

where IDENT is a dummy parameter that will be referred to throughout the 1st through Nth statements (stmt#1 . . . stmt#N). IDENT must be a valid assembly language symbol as defined by the macro assembler that you use. The character list, char_list, is a sequence of characters (e.g., CHARACTERS) ending in a space, tab, carriage return, or other assembler terminator.

The sequence of instructions is repeated once for each character in char_list. On each sequential expansion, IDENT is substituted with the next character in char_list. Thus, on the initial macro expansion, IDENT is associated with the first character of char_list, etc.

As an example, let's use IRPC . . . ENDM to generate a table of the form:

```
BYTE:     (filled with character
            given to IRPC)
ADDRESS:  (filled with zero,
            presumably to be used
            by the program later.)
```

Such a macro would have the form:

```
IRPC   IDENT,ASDF
DB     '&IDENT' ;; quote character
DW     Ø  ;; provide address space
ENDM
```

This particular macro would produce code of the form:

```
DB      'A'
DW      Ø
DB      'S'
DW      Ø
DB      'D'
DW      Ø
DB      'F'
DW      Ø
```

Note that the identifier IDENT is preceded by an ampersand ('&') inside the quoted string. The ampersand is the concatenation operator for the macro assembler and is used when we want the value of an identifier to be concatenated to other text in the macro body. If the ampersand is not present, no substitution of IDENT will occur, and the following would result for the above macro request:

```
DB      'IDENT'
DW      Ø
DB      'IDENT'
DW      Ø
DB      'IDENT'
DW      Ø
DB      'IDENT'
DW      Ø
```

Substitution would likewise not be performed if IDENT were not written in capitalized form within the quoted string. Substitution of identifiers into quoted strings is done routinely in macro instructions. Introduction at this stage is natural, since the concept will be used repeatedly.

IRP . . . ENDM:

The final in-line macro group is IRP . . . ENDM. IRP stands for "indefinite repeat." Unlike IRPC, which took two parameters (IDENT, char_list), IRP takes an identifier followed by

CP/M '83

AN INTERNATIONAL EXPOSITION AND CONFERENCE FOR THE CP/M INDUSTRY AND THE CP/M USER

FRIDAY-SUNDAY, JANUARY 21-23, 1983 MOSCONE CENTER, SAN FRANCISCO

CP/M '83 is an international exposition and conference for CP/M users, manufacturers, independent software developers, OEM's, venture capitalists, software publishers, distributors, and dealers. The exposition portion of the event will be the largest presentation of CP/M based hardware and software ever assembled. Hundreds of exhibits will showcase the full spectrum of application packages, development aids, peripherals, accessories, publications, and services for CP/M based computers.

The seminar and conference program will include noted leaders from the software industry such as Gary Kildall, Sol Libes, Adam Osborne, Tony Gold, Chris Morgan and John Rowley, who will conduct informative discussions exploring CP/M applications, technical information, development aids, venture capital programs and software distribution. End user workshops will show users how to get the most from their CP/M computer.

CP/M '83 is sponsored by Digital Research, Inc., the creators of CP/M. Over 650 different companies support CP/M and more than 5,000 companies produce CP/M application packages . . . Most of these firms will be represented at CP/M '83 . . . So attendees will see and try out application packages for every profession, plus state-of-the-art programs for word processing, telecommunications, graphics, and data base management. Also on display or as workshop subjects are development aids to help you program faster, plus hundreds of CP/M compatible products appropriate for your applications.

At CP/M '83 you can explore the entire world of CP/M under one roof and learn more in three days that you could in six months.

Call or write Northeast Expositions for the schedule of seminars, workshops and conferences and San Francisco hotel information reservation forms.

Registration is $10 for one-day exhibits-only ticket, or $20 for a three-day exhibits ticket which includes admission to workshops, seminars and conferences. To register send check (payable to Northeast Expositions) with return address.

CP/M '83

Sponsored by Digital Research, Inc.
Friday-Sunday
January 21-23, 1983
Moscone Center, San Francisco
Show Hours: 11 am to 6 pm daily
Seminars (for the trade) 9 am to 11 am
User Conferences & Workshops 11 am to 5 pm

For information on exhibiting, call or write
Northeast Expositions, Inc.
824 Boylston Street, Chestnut Hill, Massachusetts 02167
Telephone: 617-739-2000.

a variable number of parameters. Each parameter is a string of characters. These sequences have the following form:

```
lbl1:   IRP      IDENT,strl,
                 str2,...,strM
        stmt#1
        stmt#2
        ...
        stmt#N
lbl2:   ENDM
```

where, as before, 1bl1 and 1bl2 are optional labels. IDENT is the identifier that is associated with each of strl . . . strM in turn.

IRP . . . ENDM could be used to generate a sequence of null terminated strings. Here is an example:

```
IRP     IDENT,NOW,IS,THE,TIME
DB      '&IDENT',Ø
ENDM
```

This sequence would produce:

```
        DB      'NOW',Ø
        DB      'IS',Ø
        DB      'THE',Ø
        DB      'TIME',Ø
```

We have used a quoted string here and have included the required ampersand, as in the previous example.

Suppose we had wanted to include a space, comma, or other assembler delimiter as part of one of the above parameters. This would require some type of quoting mechanism. Such a mechanism is provided by angle brackets (< . . . >). Placing a string of characters in angle brackets causes it to be handled as a single parameter. One set of angle brackets is stripped off each parameter at evaluation so they won't appear in the resulting instructions or data.

Here is an example of such a parameter used in a macro similar to the above IRPC . . . ENDM sequence:

```
IRP   IDENT,<THIS IS BRACKETED>,
        LAST,STRING
DB    '&IDENT',Ø
        ENDM
```

which would produce:

```
DB    'THIS IS BRACKETED',Ø
DB    'LAST',Ø
DB    'STRING',Ø
```

We have now discussed the in-line macro facilities common to most CP/M macro assemblers. We now go on to discuss the stored macros.

Stored macros

Stored macros give a great deal of power to the assembly language programmer. Unlike in-line macros, stored macros are user-defined. Stored macros may include in-line macros containing conditionally assembled code, as well as calls to other stored macros. Recursive macro processing is also possible, but is used less often. Stored macros are defined in the following way:

```
NAME   MACRO      DPARM#1,
       DPARM#2,...,DPARM#M
       stmt#1
       stmt#2
       ...
       stmt#N
lbl1:  ENDM
```

where NAME is the name of the macro and 1b11 is an optional label. NAME must generally be different from that of any other symbol defined. DPARM#1 . . . DPARM#N are the dummy parameters used within the macro body. In a given invocation of a macro, some or all of these may be missing.

Stored macros are invoked in the following way:

```
lbl:  NAME  P1,P2,...PN
```

where lbl is an optional label, and P1 . . . PN are the actual parameters that replace the dummy parameters DPARM#1 . . . DPARM#N specified in the macro definition. Note that the values of such parameters are ASCII strings—not numeric values.

Since stored macros may contain calls to other macros (both in-line and stored) as well as conditional assembly requests, they may be quite complicated. As a simple example, let's consider a pair of macros, one that loads the BC, DE, HL registers with three consecutive words in memory, and a complementary macro that stores the BC, DE and HL registers into three consecutive memory words. The first would be as follows:

```
LOAD3   MACRO   ADDRESS
        LHLD    ADDRESS
        MOV     B,H
        MOV     C,L
;; first word to bc
        LHLD    ADDRESS+2
        XCHG
;; second to de
        LHLD    ADDRESS+4
;; third to hl
        ENDM
```

The second macro would be as follows:

```
STOR3   MACRO   ADDRESS
        SHLD    ADDRESS+4
;; third word from hl
        XCHG
        SHLD    ADDRESS+2
;; second from de
        XCHG
        PUSH    H
;; save HL
        MOV     H,B
        MOV     L,C
        SHLD    ADDRESS
;; first from bc
        POP     H
;; recover HL
        ENDM
```

These macros might be used in a program fragment of the following form:

```
        LOAD3   INPUT
        ; get bc, de, hl
        ;
        ; program performs some
        ; manipulations on
        ; this data...
        ...
        ;
        STOR3   OUTPUT
        ; store the results
        ...
INPUT:  DS      6
OUTPUT: DS      6   ; the data
```

We have now introduced the in-line and stored macro types, and given examples of each. Two auxiliary statments (LOCAL and EXITM) are used in conjunction with macros, and will be considered next.

LOCAL

After a MACRO definition line, one or more LOCAL statements may appear. The LOCAL statements contain names of variables that are to be local to the given macro expansion. That is, the scope of those symbols is confined to that macro and is not defined elsewhere. The MAC macro assembler replaces such symbols with symbols of the form ??XXXX, where XXXX is a number from 1 to 9999. Here is an example where the LOCAL statement is essential:

```
CMP16   MACRO   REG
;; compare word at hl with zero
        LOCAL   DONE
;; where to go when done
        INR     M
        DCR     M
;; set zero flag for first byte
        JNZ     DONE
;; not zero, goto done...
        INX     H
        INR     M
        DCR     M
        DCX     H
;; DCX doesn't affect Z flag
DONE:
        ENDM
;;  end of macro, Z flag
;;  set accordingly
```

If DONE could not be local, the macro could be used only once, since the symbol DONE would become multiply defined on the second invocation of CMP16.

EXITM

The EXITM statement causes the current macro expansion to be prematurely aborted. It is normally used within conditional statements (IF . . . ELSE . . ENDIF). The following example uses EXITM to terminate a macro expansion after 26 iterations. Notice the use of the concatenation operator ('&') in creating symbol names:

```
CTLSYM     MACRO
           LOCAL   COUNT
COUNT      SET     1
           IF      COUNT EQ 27
           EXITM
           ELSE
CTRL&COUNT EQU     'A'+COUNT-1
           ENDIF
           ENDM
```

This sequence produces the symbols CTRL1 . . . CTRL26, representing the 26 control characters A . . .^Z.

Conclusion

In this installment of CP/M Bus we have discussed the basics of macro instructions as supported by Digital Research's MAC and other macro assemblers. Macro sequences may be used to produce powerful working environments, including specialized languages for dedicated applications. Future columns will include a discussion of the use of macros for specific purposes in the CP/M environment, including the use of some special macro libraries provided by Digital Research with their MAC assembler.

Many CP/M users who are quite at home in 8080/Z80 assembly language stay away from macros. A frequent reason is that most macro assembler manuals are excellent reference books but poor tutorials. Hence I welcome Anthony Skjellum's simple, clear explanations, which I hope will get you interested in experimenting. "The CP/M Bus" is a forum—give us feedback in the form of questions, comments, and suggestions for other topics.

—Chris Terry

Introducing...

64K CMOS STATIC RAM BOARD

The Electralogics 64K CMOS Static Ram Board is a versatile and reliable data storage device aimed at the 8 bit micro user. "CMOS" static ram means low power consumption and inherently greater reliability and data integrity. The Board is available assembled and tested only with 64K of Ram (special orders on request) at extremely competitive prices.

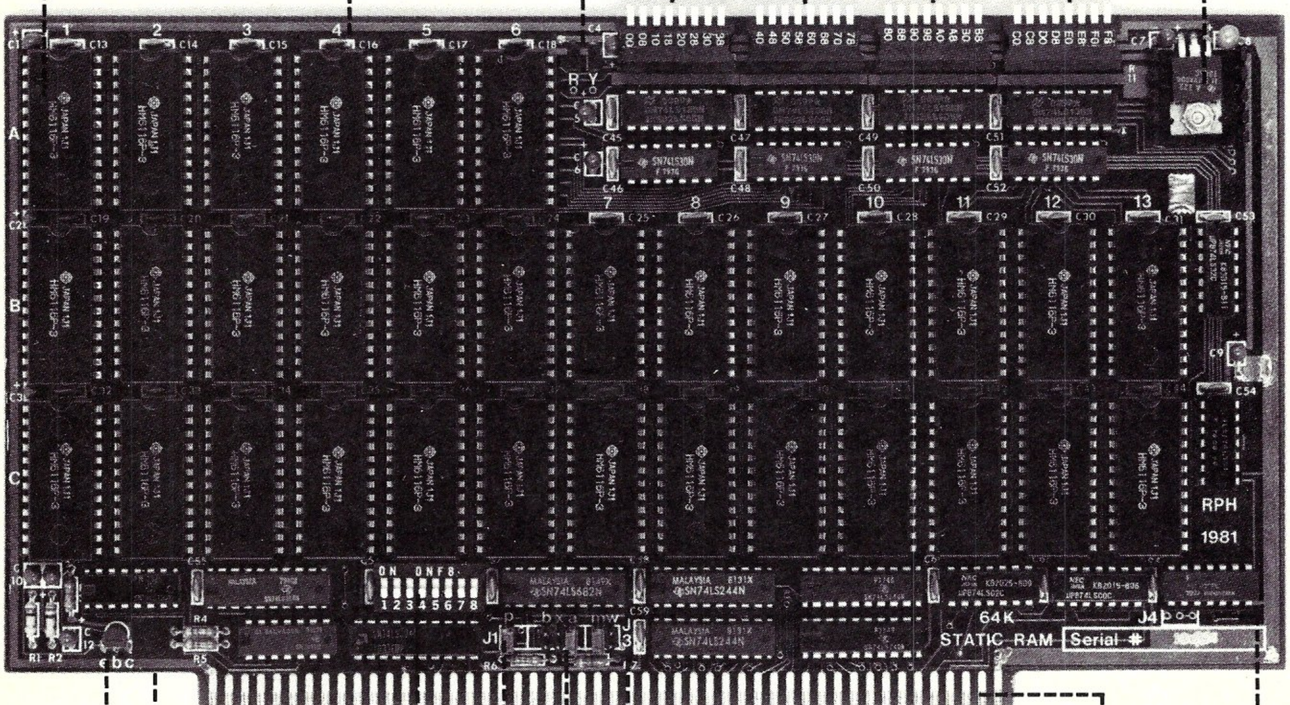
CMOS STATIC RAM (150 nsec)
- Pin for pin compatible with 2716 EPROMS.

EXTENSIVE DECOUPLING

ACTIVITY INDICATORS
- Continually indicate board status.
- Yellow – on to indicate that board is selected;
- Red – on to indicate access to board.

WINDOW SELECTION
- 32 separate switches each control a 2K window in the boards 64K address space. On = enabled.

LOW POWER REQUIREMENTS
- Low power means less heat and improved reliability.



WRITE BUFFER
- Buffers pin 21 of all Rams.

ADDRESS SELECTION
- Switches are used to set extended address and bank address.

UL APPROVED BOARD

PHANTOM ENABLE

ADDRESSING MODE SELECTION
- a – always enabled.
- x – extended addressing.
- b – Bank select.

WRITE SELECTION
- Allows writing to be gated by PWR (pin 77) or MWRT (pin 68) for use with front panel systems.

S-100/696 BUS STANDARD

FRONT PANEL ENABLE

PRICE – $499.00

Electralogics™ — Incorporated —

Manufactured by:
ELECTRALOGICS INCORPORATED
Available in Canada

CIRCLE 106 ON READER SERVICE CARD

Represented by:
INTEGRAND
8620 Roosevelt Ave.,
Visalia, California 93291

Introducing...

ELECTRALOGICS' MFIO

The most versatile and capable I/O board available for the S-100 bus.

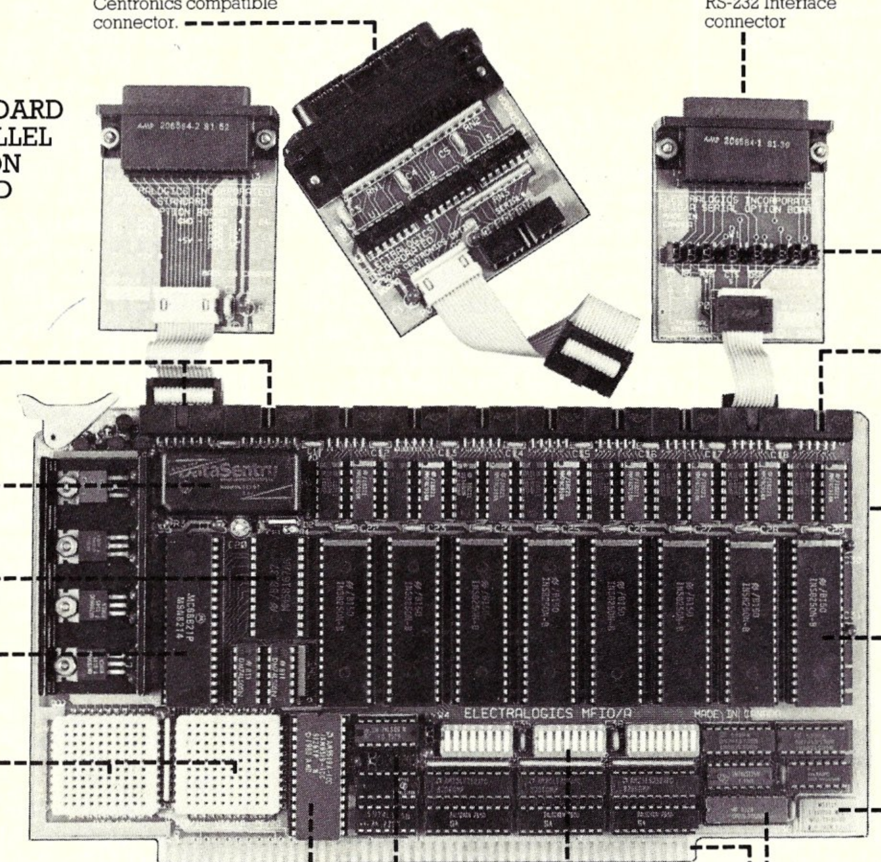
Electralogic's MFIO is the most versatile and capable I/O board available for the S-100 bus. The 8 asynchronous serial ports, 2-8 bit bidirectional parallel ports, 8 level programmable interrupt controller and battery backed-up real time clock provide all the features which traditionally required 3 or more boards.

The design meets the needs of OEM's and system integrators who demand high density and reliability in their products.

Additional capabilities include: extended I/O addressing, up to 6 wait states, jumper selectable for high speed systems, easy to use interface cards and serial data rates up to 57.6 K baud.

The 2 pin boards allow any of 20 interrupt sources (11 on board + 9 from S100 bus) to activate 1-8 interrupt levels. The board comes complete with extensive manual and source listings for standard CPM* BIOS, interrupt driven BIOS, clock set routine, time print routine, diagnostic routines and sample device initialization routines.

*CPM is a trademark of DIGITAL RESEARCH



PRICE – $699.00

CENTRONICS OPTION BD. – $39.95
SERIAL OPTION BD. – $24.95
STANDARD PARALLEL OPTION BD. – $24.95

Electra™ logics — Incorporated —

Manufactured by:
ELECTRALOGICS INCORPORATED
Available in Canada

CIRCLE 106 ON READER SERVICE CARD

OEM and Dealer Inquiries Invited

Represented by:
INTEGRAND
8620 Roosevelt Ave.,
Visalia, California 93291

Letters to the Editor

Dear Editor,

I am a new subscriber to *Microsystems*, and I was delighted that the very first issue repaid my subscription with the article on the "Hardware Random Byte Generator" by John Gillespie. Since I frequently employ random numbers in my computer simulation work, I constructed the generator as described for my 2MHz Z80 system. However, I found that with the 10MHz clock for the generator, the output was sometimes in transition during the read. However, a 5MHz clock works fine. The 10MHz crystal can still be used if the other half of the 74LS74 is used as a divide-by-two (clock input to C, Q* connected to D, output from Q). The divide-by-two circuit is inserted before feeding the 10 MHz clock to the last inverter and the 74LS164 clock inputs.

Alan D. Howard
Rt. 3, Box 680
Crozet, VA 22932

Dear Editor,

As a user of Allen Ashley's PDS system on North Star for the past 4 years, I would like to correct an inaccuracy in the review of this package in the Jul/Aug 1982 issue.

In the North Star DOS version, it is *not* necessary to know the size of EDIT's output file in advance. You merely need to create a file with size zero; then when EDIT saves your text to the file, it will automatically compute the size and update the directory accordingly (it adds an extra block for future expansion). Of course, it will not change the size of an already existing file if you want to expand that file, but this is a failing of North Star's operating system, which allocates disk sectors sequentially, and not the fault of PDS, which is constrained to operate within the limits imposed by the operating system.

Also, you didn't mention that you also get an enhanced version of EDIT, called PROTEXT, which includes all the capabilities of EDIT and provides, in addition, a limited text formatting capability (similar in operation to a simple RUNOFF program). This is a new program, so perhaps the copy you received for review did not include it.

In any case, I can say without hesitation that PDS has been the best software buy I ever made for my North Star system. If you are interested, Mr. Ashley also sells a disassembler, a hybrid development system (North Star Basic—assembly language development system interface), and a compiler for North Star Basic.

Mike Shefler
Zorph Enterprises
3646 Gibsonia Rd.
Gibsonia, PA 15044

Dear Editor,

A tribute to the correspondence columns of *Microsystems*. After my letter about the Big-Z CPU board, I had several most helpful letters. By then, I had actually made the thing go, but the modifications suggested by Jim Warner of the University of California at Santa Cruz have made it go better.

A proposal. What about a design for home construction of:

1. A memory board, 64K static preferably but dynamic if the cost of static chips is still prohibitive.
2. A CPU board with onboard serial interface and a monitor ROM which can be automatically removed from memory space.

Both of these should be *strictly* IEEE "S-100"-conforming, and properly documented both with regard to construction, operation, and fault-finding. They need not cater to all the variants that most commercially produced boards have to, but would form the basis of a good S-100 system that could be relied on to function properly.

Suppliers of kits or assembled board would have to undertake to *Microsystems* to furnish post-supply service and advice. This sort of thing is popular in England, but doesn't seem to have been tried much in the U.S.

Prof. P. F. Ridler
Head: Computing Science
University of Zimbabwe
P.O. Box M.P. 167
Mount Pleasant,
Salisbury, Zimbabwe

Dear Editor,

Wordcraft has been selling a communications package, THE MICRO LINK©, for about a year and a half. It has been popular on the Osborne 1 and other computers, and we now have an IBM-PC program.

Recently Lifeboat Associates began advertising a package called Micro Link-80.

People who tried to purchase from Lifeboat say that the firm is unwilling to take orders or unable to fill them after three-month delays.

We're letting the CP/M and CP/M-86 community know that there is only one program called THE MICRO LINK©, and Lifeboat does not distribute it. (Digital Marketing does.)

Our lawyer has informed Lifeboat of the apparent infringement of our copyright.

Now that we have just issued THE MICRO LINK II©, a completely revamped program, Lifeboat's confusion of the matter becomes even more unfortunate.

Geoffrey Sinclair
Wordcraft
3827 Penniman Ave.
Oakland, CA 94619

Sir,

I wish to discuss a few statements made by Mr. Klossner in his article, "Cloning CP/M Disk Drive," on pages 70 to 72 of the July/August 1982 issue.

First: By common convention the storage location of the current disk number is hex 0004 in a standard CP/M system, as mentioned in the *CP/M Alteration Guide*, page 23. It might

JANUS

Question: When can I use Ada?

ADA*

Answer: NOW — with **JANUS.**

Ada is available now for your micro-computer.

JANUS is a subset of **Ada** which includes those features sorely missed by programmers on micros. Here is a list to help you decide for yourself.

+ Modular Separate Compilation
+ Single and Double Precision Floating Point Numbers
+ Binary Coded Decimal (BCD) Fixed Point Numbers
+ Integer and Long Integer Numbers
+ Complete String Handling
+ Sequential and Random Access 1/0
+ Full Dynamic Allocation and Deallocation
+ Friendly Error Handling
+ An Assembler for interfacing assembly routines
+ A Linker for combining modules
+ True native code is produced
+ ROMable, reenterant code
+ Run-time library source code
+ Low Cost. **JANUS** is more cost effective than any other comparable Ada package
+ Inexpensive Updates
+ No royalties for programs written in **JANUS**
+ No hassle customer service

The language that is based on the past but looks to the uses of the future.

In short, all pluses. **JANUS** contains everything you need to do fast, structured program development in a micro environment.

JANUS is available for the CP/M, CP/M-86, and MS-DOS operating systems.

Now you too can take a step forward into the future on these computers:

8080/Z80 based systems: (All CP/M) Apple Softcard, North Star, Cromemco, Superbrain, TRS-80 Model II, and all CP/M 8″ disk systems.

8086 based systems: IBM Personal Computer, Victor 9000, Seattle Computer System II, Tecmar, Lomas Data Products, and all CP/M 8″ disk systems.

8080 or Z80, CP/M (requires 56K memory) — $300.00

8086/8088, CP/M-86 or MS-DOS (requires 96K memory) — $400.00

CP/M, CP/M-86, MP/M-86 are trademarks of Digital Research, Inc.
* ADA is a trademark of the U.S. Department of Defense
MS-DOS is a trademark of Microsoft

86-DOS is a trademark of Seattle Computer Products
SB-86 is a trademark of Lifeboat Associates
Apple Softcard is a trademark of Microsoft, Inc.

RR SOFTWARE

specialists in state of the art programming

P.O. BOX 1512 MADISON, WISCONSIN 53701

(608) 244-6436

be wise to adhere to this.

Second: The remarks on page 71 seem to indicate that Mr. Klossner confuses CP/M "block size" with "sector size." The first is a logical unit used by Digital Research (one easily admits they don't use it too clearly in their manuals), whilst the second is a physical size used in hardware and hardware-dependent parts of software. (From hereon read "BDOS" for "CP/M.")

It is my understanding that CP/M *never* reads or writes anything but "records" that happen to correspond to 128 bytes and therefore, miracle of miracles, to a sector of standard size on a single-density 8″ floppy. However, even while CP/M may internally use allocation block sizes of 1024 bytes to 16,384 bytes, and while the former may correspond to sector size available on double-density formatted disks, CP/M never writes anything but 128-byte "records" to the CBIOS.

Buffering, if any, must be done by the CBIOS which, as far as CP/M is concerned, is at liberty to buffer as many "records" as it likes into sectors in its choice of size for actual hardware disk I/O.

Third: CP/M therefore does not "flush" any buffer. This is done by the CBIOS, which it is my understanding Mr. Klossner did not mean when he alluded to CP/M.

And any well-designed CBIOS would, of course, to continue Mr. Klossner's scenario, either keep track of which part of its buffer belonged to which drive or, rather, flush its buffer appropriately before selecting a new disk.

This can easily be accomplished by introducing a level of CP/M sector to "host sector" translation between the CBIOS entry level and the actual controller firmware/hardware level (see App. G to the CP/M Alteration Guide for a skeletal idea about this).

Lastly: In my experience, come at by hacking on a single-drive system for the last two years, the most frustrating time is trying to PIP from one drive to another and having to swap floppies every other record or so. To this there is a remedy!

Get yourself a copy of the "Multiple File Transfer" program, published in the October 1980 copy of *Dr. Dobb's Journal*. Or get it from the CPM User Group at N.Y. on their Volume 63. This cutie reads as many files as will fit the TPA of CP/M, then asks you to insert the destination disk *without* selecting a different drive and writes them back onto this diskette. The only thing you may have to beware of are the different densities of your diskettes. The original (3.0) published version didn't work when I typed it in. I now have a version of my own that does. How-

FMS
The Software Machine
Powerful Fast Responsive

Performance, speed, control, ease of use. That's what you expect from a finely tuned machine. And that's what your data management software should deliver too. The *new* FMS-80 Version 3 gives you this and *more*–a fully integrated Applications Development System that makes even the most complex application easy.

Almost everyone needs to manipulate information. With FMS Version 3 even a beginner can follow the simple menu selections and be off and running in almost no time. Customized screens and user menus are easy to design. Powerful full-screen editing makes entering, modifying, adding or deleting data a snap.

FMS makes getting your information out easy too. Interactive QUERY and comprehensive SELECT can extract the data you need almost instantly. Our powerful Report Generator can produce almost any imaginable report with minimal effort. FMS takes you by the hand each step of the way.

More Than Just a Database Manager.

If you've been around the track a few times already, FMS is for you too. Our enhanced Version 3 EFM programming language gives you total control. Our ISAM-like multi-key data structure, access to 19 open files, full string handling, alphanumeric variable, field and file names, 18 digit FP&BCD math, structured programming constructs and other advanced features make EFM the language of choice for data management applications. FMS can make you more productive and save you time and money, whether you're developing a simple mailing list or a complex turn-key general accounting system.

The UNIX-inspired FMS Shell brings advanced capabilities like command stream manipulation and dynamic input and output redirection to the CP/M world for unprecedented control of the operating environment.

Don't Run Out of Gas.

FMS's capabilities go way beyond other data manipulation programs. More fields per record, more open files, more variables, more everything.

	FMS	dBASE	Condor
Maximum fields per record	255	32	127
Maximum number of variables	281	64	0
Maximum number of open files in a program	19	2	2
Maximum number of open files in a report	19	1	1
Maximum display pages per record	255	3	1

Don't lock yourself into a system that can't handle the big jobs!

A Proven Winner

FMS-80 has been leading the field since 1978. Now Version 3 sets new standards for the future. Contact your local dealer for a test drive.

DJR Associates, Inc.
303 S. Broadway • Tarrytown, N.Y. 10591
(914) 631-6766 • Telex 646792 DJR NTAR

CIRCLE 22 ON READER SERVICE CARD

ever, since I have not seen the source code of the CPMUG version 4.5, maybe that's fixed by now.

I may add that a number of programs which, by their description, do the very same thing are sold for money—not my money, though . . .

Joachim Wetzig
Bacchusring 9
D 6500 Saulheim
West Germany

Andrew Klossner replies:

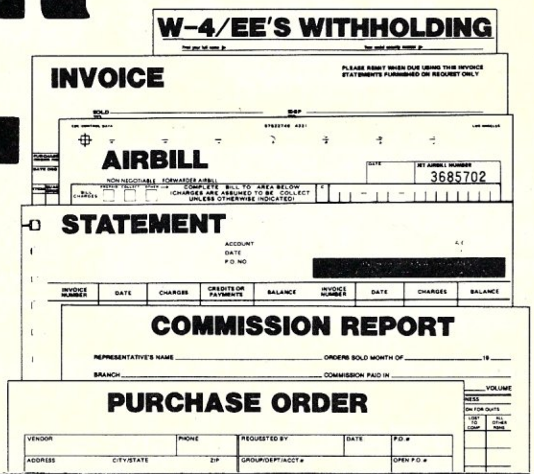
The low four bits of location 0004 contain the "currently logged-in" disk, not the disk that is currently active. The currently logged-in disk is the one that appears in the command prompt, e.g., "A>." Therefore a different byte must be used to remember which disk was last the target of an I/O request.

CP/M does in fact flush buffers. The interested reader should refer to chapter 12, "Sector blocking and deblocking," of the *CP/M 2.0 Alteration Guide*, which describes the mechanism by which the BDOS instructs the CBIOS to flush its buffer. This buffer contains only one physical sector at a time, and is needed because disk interfaces typically do not allow the CBIOS to read or write less than an entire sector.

There are many well-designed CBIOSs that do not flush the physical sector buffer when selecting another disk. For example, if disk A is double density with 512 bytes per sector and disk B is single density with 128 bytes per sector, an optimal CBIOS will not flush the internal buffer when switching from A to B to read a record, since that record need not be deblocked and can be transferred directly to the program's DMA.

Inserting a translation between the CBIOS entry level and the actual controller level is exactly the technique of the second method presented in the article. The important point is that any modification to the

END THE PAPER CHASE.



Now you can computerize your business forms and input screens without all that tedious, time consuming programming.

With ZIP,™ it's going to seem almost too easy.

Just "paint" the screen.

To prepare an input screen or output form, just move the cursor around the screen and type in text, prompts and data field names exactly where you want them. Use "@" to print or display values, use ";" for input fields.

When the screen looks like the format you want, type "/S" and what you see is what you'll get. In seconds, ZIP writes all the program code to recreate the format on the screen or on paper.

And you can use the ZIP code immediately just by adding a single line to your main program: GOSUB xxxxx in BASIC, DO Formname in dBASE II.

ZIP is quick and friendly.

ZIP runs on virtually every 8-bit micro known to man, and your terminal continues to work the way it did (tab, arrows, etc.), so you don't have to learn or unlearn anything about your equipment.

Commands are typed (no control codes), so you or your secretary can breeze through input screens and output forms up to 88 lines long and have ZIP whip out the BASIC or dBASE II code.

The ZIP Talker,™ a line at the bottom of the screen, always tells you exactly where you are. And Help is just two keystrokes away.

Now MBASIC really ZIPs.

The MBASIC version goes further and gives your programs the same screen handling characteristics that ZIP has, by writing a piece of itself in MBASIC so that you can use it in any of your programs.

The operator can use the arrows, etc. during data entry and conveniently jump back and forth between the input fields.

You can specify field lengths, or let ZIP default to the available space. Either way, text and prompts are protected no matter what kind of terminal you have, so the operator can't write over the fields and prompts.

```
              * * * * * * * * * * * * * * *
                    NEW CUSTOMER
              * * * * * * * * * * * * * * *

  CUSTOMER NAME:  ;Customer$              DATE: @Date
   MAIL ADDRESS:  ;Addr1$:30
           CITY:  ;Addr2$
          STATE:  ;Addr3$     ZIP:  ;Zip1:5
   SHIP ADDRESS:  ;Addr4$:30
           CITY:  ;Addr5$
          STATE:  ;Addr6$     ZIP:  ;Zip2:5
DISCOUNT CATEGORY:  ;Rate!
THE CODE FOR THIS NEW CUSTOMER IS @CustCode.

Row 17, Col 36
```

You get the MBASIC code for a "Talker" that you can use to pretty up your program prompts.

And easy, one-line data validation is built in.

Join thousands of users ZIPping along.

All you need is an 8-bit micro with CP/M or MPM, 48k of memory and a 24x80 ASCII or ANSI terminal (Osborne 1 and 56k Apple okay, too).

The MBASIC and CBASIC versions are $160 each ($225 for both) plus $7 shipping (VISA, MasterCharge or money order). The dBASE II version is available alone from Ashton-Tate (213-204-5570), or we'll sell you dBASE II with ZIP for $650. For more information, contact Nexus, 5455 Wilshire, Suite 802, Los Angeles, CA 90036.

Or if you'd like to end the paper chase sooner, just call 800-227-3747. (In California, call 213-937-0554, add 6% tax.)

ZIP™ from Nexus
The man-machine connection.

Letters continued . . .

CBIOS to cause it to support multiple virtual drives must hook into the blocking/deblocking mechanism.

The latest version of PIP, as distributed with CP/M version 2.2, will not make the operator swap floppies at every other record. Instead, it loads up all of available memory with multiple records from the input file before writing any records to the output file. The Multiple File Transfer program is adequate for the task of transferring files between disks, but does not address the problem of allowing a standard CP/M program to access files on several drives when only one drive is available. To achieve this capability, the CBIOS modifications outlined in the article must be made.

Andrew Klossner
PO Box 283
Wilsonville, Oregon 97070

Dear Editor,

The reason I'm writing is the article about the PDS system in the July/August issue, page 36. I have been using PDS for almost four years now, and Mr. Zeiger's article is the first one I have seen about Mr. Ashley's fine set of programs. I would like to add a few comments, and maybe attempt to clear up a few things. When I upgraded my PDS from single to double density, Mr. Ashley threw in a text formatter/editor called PROTEXT, which he said is going to be on all PDS disks now. This letter was written using PROTEXT to format the printout. Also, for a mere $25, you can get the REGENT disassembler, which produces a raw disassembly file for any program in RAM. You can't specify any data areas like in RESOURCE, but REGENT places the hex code into the disassembly file in ASCII. If that hex code evaluates to an alphabetic character, that character is substituted for the hex code. That makes it easy for you to find any messages embedded in the program. I have

Letters continued . . .

used it on many occasions, disassembling programs as large as 18K.

Now a few corrections to the article: Mr. Zeiger stated that, when you use ASMB or EDIT (North Star DOS versions only) and your source file gets bigger than the old file, you will have to go back to DOS, create the file, and then save it. *Not so*! All PDS modules producing an output file can dynamically allocate disk space, IF the output disk file is a zero length file. For example, you have the source file XXXX with a length of 100 blocks and you expect it to grow bigger during your work. Simply create a file, for example "XXXX1" with size 0 at the same disk address as "XXXX". Use XXXX1 as output file. When you're done and save your file back to disk, XXXX1 will be set to the appropriate size by the PDS module you are using. (REGENT and PROTEXT have the ability to create a file if it does not exist on disk).

A drawback not mentioned by Mr. Zeiger is EDIT's inability to recognize TAB stops. While you can enter ^I into a file, no PDS module is prepared to recognize it as a text formatting character.

I have to agree with Mr. Zeiger on the value of PDS for the North Star system. I am currently using CP/M on my North Star machine, but do not have PDS for CP/M yet. I expect to get it in a few weeks and until then I'll have to keep using ED and ASM, both of which are a nuisance and very cumbersome to use if you're used to better things. I am looking forward to using PDS on CP/M and I'm convinced it's the best buy at $99 (or at the upgrade fee to go from N* to CP/M). I think only the lack of advertising for the product has kept it from being more widely used.

Siegmund F. Kluger
11025 Vista Del Sol #250-II
El Paso, TX 79935

Disk Drive for Apple $289.95

Modems

SIGNALMAN - Anchor
Direct-connect automatic answer/originate selection, 300 Baud full duplex, Bell 103, includes RS-232 cable

IOM-5600A	Signalman	$89.95

SMARTMODEM - Hayes
Sophisticated direct-connect auto-answer/auto-dial modem, touch-tone or pulse dialing, RS-232C interface, programmable

IOM-5400A	Smartmodem	$224.95
IOK-1500A	Hayes Chronograph	$218.95
IOM-2010A	Micromodem II	$328.95
IOM-2012A	Terminal program for MMII	$89.95
IOM-1100A	Micromodem 100	$368.95

APPLE-CAT - Novation
Software selectable 1200 or 300 Baud, direct connect, auto-answer/auto-dial, touch & pulse dialing, auxiliary 3-wire RS-232C serial port for printer

IOM-5232A	Save $50.00	$324.95

1200 BAUD SMARTMODEM - Hayes
1200 and 300 baud, all the features of the standard Smartmodem plus 1200 baud, 212 compatible, full or half duplex.

IOM-5500A	Smartmodem 1200	$599.95

1200 BAUD AUTO CAT - Novation
212 Auto Cat, 1200 & 300 baud, auto dial/answer/disconnect, LED readout displays mode, analog/digital loop-back self tests, usable with multi-line phones.

IOM-5231A	212 Auto Cat	$649.95

Video Monitors

HI-RES 12" GREEN - Zenith
15 MHz bandwith 700 lines/inch, P31 green phosphor, switchable 40 or 80 columns, small, light-weight & portable.

VDM-201201	List price $189.95	$129.95

12" GREEN SCREEN - NEC
20 MHz bandwidth, P31 phosphor ultra-high resolution video monitor with audio.

VDM-651200	Deluxe model	$199.95
VDM-651260	Economy model	$149.95

12" COLOR MONITOR - NEC
High resolution color monitor with audio.

VDC-651212	Color monitor	$389.95
NEC-1202D	RGB color monitor	$999.95

13" COLOR MONITORS - BMC
18 MHz RGB & composite video color monitors.

VDC-421320	13" RGB Color	$329.95
VDC-421310	13" Composite video	$299.95
VDX-420090	RGB card for Apple	$149.95

COLOR MONITORS - Amdek
Reasonably priced color video monitors.

VDC-80130	13" Color I	$379.95
VDC-801320	13" Color II	$894.95
IOV-2300A	DVM board for Apple	$199.95

AMBER or GREEN MONITORS - Jade
High resolution 18 MHz compact video monitors.

VDM-751210	12" Amber phosphor	$149.95
VDM-751220	12" Green phosphor	$139.95
VDM-750910	9" Amber phosphor	$149.95
VDM-750920	9" Green phosphor	$139.95

EPROM Erasers

ULTRA-VIOLET EPROM ERASERS
Inexpensive erasers for industry or home.

XME-3100A	Spectronics w/o timer	$69.50
XME-3101A	Spectronics with timer	$94.50
XME-3200A	Economy model	$39.95

Single User System

THREE BOARD SET - SD Systems
4 MHz Z-80A CPU, 64K RAM (optional 256K), serial I/O port, parallel I/O port, double density disk controller, CP/M 2.2 & manual set, system monitor, control & diagnostic software. Includes SBC-200, 64K ExpandoRAM II, Versafloppy II, & CP/M 2.2 - all boards are assembled & tested.

Board set with 64K of RAM	$1095.00
Board set with 256K of RAM	$1295.00

Apple II Accessories

APPLE DISK DRIVE - Fourth Dimension
Totally Apple compatible, 143,360 bytes per drive on DOS 3.3, half-track capability - reads all Apple software, plugs right in to Apple controller as second drive, DOS 3.3, 3.2.1, Pascal, & CP/M compatible.

MSM-123200	40 Track add on Apple drive	$289.95
MSM-123200	Controller with free DOS 3.3	$99.95

16K RAM CARD - for Apple II
Expand your Apple II to 64K, use as language card, full 1 year warranty. Why spend $175.00 ?

MEX-16700A	Save over $100.00	$69.95

Z-80 CPU CARD - for Apple II
Two computers in one, Z-80 & 6502, more than doubles the power and potential of your Apple, includes Z-80 CPU card CP/M and complete manual set.

CPX-62800A	A & T with software	$249.95

APPLE-CAT - Novation
Software selectable 1200 or 300 baud, direct connect, auto-answer/auto-dial, auxiliary 3-wire RS232C serial port for printer.

IOM-5232A	Save $50.00!!!	$325.95

8" DISK CONTROLLER - Vista
New from Vista Computer, single or double sided, single or double density, compatible with DOS 3.2/3.3, Pascal, & CP/M 2.2. Shugart & Qume compatible

IOD-2700A	A & T	$499.95

2 MEGABYTES for Apple II
Complete package includes: Two 8" double-density disk drives, Vista double-density 8" disk controller, cabinet, power supply, & cables, DOS 3.2/3.3, CP/M 2.2, & Pascal compatible.

1 MegaByte Package	Kit	$1495.00
1 MegaByte Package	A & T	$1695.00
2 MegaByte Package	Kit	$1795.00
2 MegaByte Package	A & T	$1995.95

VISION 80 - Vista Computer
80 column x 24 line video card for Apple II, 128 ASCII characters, upper and lower case, 9 x 10 dot matrix with 3 dot descenders, standard data media terminal control codes, CP/M Pascal & Fortran compatible, 50/60 Hz

IOV-2400A	Vista Vision 80	$299.95

CPS MULTICARD - Mtn. Computer
Three cards in one! Real time clock/calendar, serial interface, & parallel interface - all on one card.

IOX-2300A	A & T	$179.95

HI-RES GRAPHICS CARD - Genie
Intelligent printer interface and control card allows full high resolution graphics and screen dumps.

IOP-2405A	Genie for Epson	$119.95
IOP-2410A	Genie for Okidata	$119.95
IOP-2415A	Genie for NEC/C. Itoh	$119.95

Power Strips

ISOBAR - GSC
Isolates & protects your valuable equipment from high voltage spikes & AC line noise, inductive isolated ground, 15 amp circuit breaker, U.L. listed

EME-115103	3 socket	$39.50
EME-115105	4 socket	$49.50
EME-115100	8 socket	$54.50
EME-115110	9 socket rackmount	$74.50

Single Board Computer

SUPERQUAD - Adv. Micro Digital
Single board, standard size S-100 computer system, 4 MHz Z-80A, single or double density disk controller for 5¼" or 8" drives, 64K RAM, extended addressing, up to 4K of EPROM, 2 serial & 2 parallel I/O ports, real time interrupt clock, CP/M compatible.

CPC-30800A	A & T	$724.95
IOX-4232A	Serial I/O adapter	$29.95

Z-80 STARTER KIT - SD Systems
Complete Z-80 microcomputer with RAM, ROM, I/O, keyboard, display, kludge area, manual, & workbook.

CPS-30100K	Kit with workbook	$299.95
CPS-30100A	A & T with workbook	$469.95

AIM-65 - Rockwell International
Complete 6502 microcomputer with alphanumeric display, printer, keyboard, & instruction manual.

CPK-50165A	1K AIM-65	$424.95
CPK-50465A	4K AIM-65	$474.95
SFK-74600008E	8K Basic ROM	$64.95
SFK-64600004E	4K assembler ROM	$43.95
SFK-74600020E	PL/65 ROM	$84.95
SFK-74600010E	Forth ROM	$64.95
SFK-74600030E	Instant Pascal	$99.95
PSX-030A	Power supply	$64.95
ENX-000002	Enclosure	$54.95

SPECIAL PACKAGE
4K AIM-65, 8K Basic, power supply, & enclosure

Special Package Price	$649.95

S-100 EPROM Boards

PROM-100 - SD Systems
2708, 2716, 2732 EPROM programmer with software.

MEM-99520K	Kit with software	$189.95
MEM-99520A	A & T with software	$249.95

PB-1 - SSM Microcomputer
2708, 2716 EPROM board with on-board programmer.

MEM-99510K	Kit with manual	$154.95
MEM-99510A	A & T with manual	$219.95

EPROM BOARD - Jade
16K or 32K uses 2708 or 2716 EPROMs, 1K boundary.

MEM-16230K	Kit w/o EPROMs	$79.95
MEM-16230A	A & T w/o EPROMs	$119.95

S-100 Video Boards

SPECTRUM COLOR - CompuPro
Full-function color graphics board, up to 8 colors, 256 x 192 graphics, parallel I/O port, 8K RAM.

IOV-1870A	A & T	$348.95
IOV-1870C	CSC	$398.95

MICROANGELO - Scion
Ultra-high-resolution 512 x 480, 256 color or black & white S-100 video board

IOV-1500A	A & T	$799.95

S-100 MotherBoards

ISO-BUS - Jade
Silent, simple, and on sale - a better motherboard

6 Slot (5¼" x 8⅝")

MBS-061B	Bare board	$22.95
MBS-061K	Kit	$39.95
MBS-061A	A & T	$69.95

12 Slot (9¾" x 8⅝")

MBS-121B	Bare board	$34.95
MBS-121K	Kit	$69.95
MBS-121A	A & T	$109.95

18 Slot (14½" x 8⅝")

MBS-181B	Bare board	$54.95
MBS-181K	Kit	$99.95
MBS-181A	A & T	$149.95

ACTIVE TERMINATOR - CompuPro
A true mother's helper.

TSX-100A	A & T	$59.45

Prices may be slightly higher at our retail locations. Please call the store nearest you for local price and availability.

64K Static RAM $299.95

S-100 CPU Boards

8086/8087 - CompuPro
16 bit, 8 or 10 MHz 8086 CPU with provisions for 8087 & 80130.

CPU-70520A	*8 MHz 8086 A & T*	**$624.95**
CPU-70520C	*8 MHz 8086 CSC*	**$764.95**
CPU-70530A	*with 8087 A & T*	**$1224.95**
CPU-70530C	*with 8087 CSC*	**$1455.95**

8085/8086 - CompuPro
Both 8 & 16 bit CPUs, standard 8 bit S-100 bus, up to 8 MHz, accesses 16 Megabytes of memory.

CPU-20510A	*6 MHz A & T*	**$398.95**
CPU-20510C	*6/8 MHz CSC*	**$497.95**

CPU-Z - CompuPro
2/4 MHz Z80A CPU, 24 bit addressing.

CPU-30500A	*2/4 MHz A & T*	**$279.95**
CPU-30500C	*3/6 MHz CSC*	**$374.95**

SBC-200 - SD Systems
4 MHz Z-80A CPU with serial & parallel I/O, 1K RAM, 8K ROM space, monitor PROM included.

CPC-30200A	*A & T*	**$399.95**

THE BIG Z - Jade
2 or 4 MHz switchable Z-80 CPU board with serial I/O, accomodates 2708, 2716, or 2732 EPROM, baud rates from 75 to 9600.

CPU-30201B	*Bare board w/manual*	**$35.00**
CPU-30201K	*Kit with manual*	**$149.95**
CPU-30210A	*A & T with manual*	**$199.95**

CB-2 - SSM Microcomputer
2 or 4 MHz Z-80 CPU board with provision for up to 8K of ROM or 4K of RAM on board, extended addressing, IEEE S-100, front panel compatible.

CPU-30300K	*Kit with manual*	**$229.95**
CPU-30300A	*A & T with manual*	**$274.95**

2810 Z-80 CPU - C.C.S.
2 or 4 MHZ Z-80 CPU with serial I/O port & on-board monitor PROM, front panel compatible.

CPU-30400A	*A & T with PROM*	**$289.95**

2820 Z-80 DMA CPU - C.C.S.
4 MHz Z-80 CPU board with 2 serial I/O ports & Centronics parallel I/O port, separate data & status ports, DMA daisy chain compatible.

CPU-30420A	*A & T with manual*	**$569.95**

S-100 Disk Controllers

DISK 1 - CompuPro
8" or 5¼" DMA disk controller, single or double density, single or double sided, 10 MHz.

IOD-1810A	*A & T*	**$449.95**
IOD-1810C	*CSC*	**$554.95**
SFC-52506580F	*8" CP/M 2.2 for Z-80*	**$174.95**
SFC-52506586F	*8" CP/M 2.2 for 8086*	**$299.95**
SFO-54158000F	*Oasis single user*	**$499.95**
SFO-54158002F	*Oasis multi-user*	**$849.95**

VERSAFLOPPY II - SD Systems
Double density disk controller for any combination of 5¼" and 8" single or double sided, analog phase-locked loop data separator, vectored interrupts, CP/M 2.2 & Oasis compatible, control/diagnostic software PROM included.

IOD-1160A	*A & T with PROM*	**$359.95**
SFC-55009047F	*CP/M 2.2 with VF II*	**$99.95**

2242 DISK CONTROLLER - C.C.S.
5¼" or 8" double density disk controller with on-board boot loader ROM, free CP/M 2.2 & manual set.

IOD-1300A	*A & T with CP/M 2.2*	**$399.95**

DOUBLE D - Jade
High reliablity double density disk controller with on-board Z-80A, auxiliary printer port, IEEE S-100, can function in multi-user interrupt driven bus.

IOD-1200B	*Bare board & hdwr man*	**$59.95**
IOD-1200K	*Kit w/hdwr & sftwr man*	**$299.95**
IOD-1200A	*A & T w/hdwr & sftwr man*	**$325.95**
SFC-59002001F	*CP/M 2.2 with Double D*	**$99.95**

S-100 Memory Boards

256K RAMDISK - SD Systems
ExpandoRAM III expandable from 64K to 256K using 64K x 1 RAM chips, compatible with CP/M, MP/M, Oasis, Cromemco, & most other Z-80 based systems, functions as ultra-high speed disk drive when used with optional RAMDISK software.

MEM-65064A	*64K A & T*	**$474.95**
MEM-65128A	*128K A & T*	**$574.95**
MEM-65192A	*192K A & T*	**$674.95**
MEM-65256A	*256K A & T*	**$774.95**
SFC-55009000F	*RAMDISK sftwr CP/M 2.2*	**$44.95**
SFC-55009000F	*RAMDISK with EXRAM III*	**$24.95**

128K RAM 21 - CompuPro
128K x 8 bit or 64K x 16 bit static RAM board, 12 MHz, 24 bit addressing.

MEM-12810A	*A & T*	**$1609.95**
MEM-12810C	*CSC*	**$1794.95**

64K RAM 17 - CompuPro
64K CMOS static RAM board, 10 MHz, low power less than 4 watts, DMA compatible, 24 bit addressing.

MEM-64180A	*64K A & T*	**$549.95**
MEM-64180C	*64K CSC*	**$698.95**

64K RAM 16 - CompuPro
32K x 16 bit or 64K x 8 bit low power static RAM board, 10 MHz, 24 bit addressing.

MEM-32180A	*RAM 16 A & T*	**$598.95**
MEM-32180C	*RAM 16 CSC*	**$698.95**

64K STATIC RAM - SSM
IEEE 696/S-100 standard, up to 6MHz/8 Bit, 12MHz/16 Bit, 24 Bit extended addressing, disable-able in 2K increments

MEM-64300A	*A & T*	**$499.95**

64K STATIC RAM - Mem Merchant
64K static S-100 RAM card, 4 to 16K banks up to 8 MHz.

MEM-64400A	*64K A & T*	**$499.95**

64K STATIC RAM - Jade
Uses new 2K x 8 static RAMs, fully supports IEEE 696 24 bit extended addressing, 200ns RAMs, lower 32K or entire board phantomable, 2716 EPROMs may be subbed for RAMs, any 2K segment of upper 8K may be disabled, low power typically less than 500ma.

MEM-99152B	*Bare board*	**$49.95**
MEM-99152K	*Kit less RAM*	**$99.95**
MEM-32152K	*32K kit*	**$199.95**
MEM-56152K	*56K kit*	**$289.95**
MEM-64152K	*64K kit*	**$299.95**
Assembled & Tested		**add $50.00**

2066 64K RAM - C.C.S.
64K RAM board with bank and block select switching functions for Cromemeco Cromix & Alpha Micro.

MEM-64566A	*64K A & T*	**$424.95**

64K EXPANDORAM II - SD Systems
Expandable RAM board from 16K to 64K using 4116 RAM chips.

MEM-16630A	*16K A & T*	**$344.95**
MEM-32631A	*32K A & T*	**$364.95**
MEM-48632A	*48K A & T*	**$384.95**
MEM-64633A	*64K A & T*	**$399.95**

MEMORY BANK - Jade
4 MHz S-100 bank selectable expandable to 64K.

MEM-99730B	*Bare board w/manual*	**$49.95**
MEM-99730K	*Kit with no RAM*	**$179.95**
MEM-32731K	*32K kit*	**$199.95**
MEM-64733K	*64K kit*	**$249.95**
Assembled & Tested		**add $50.00**

16K STATIC RAM - Mem Merchant
4MHz lo-power static RAM board, IEEE S-100, bank selectable, addressable in 4K blocks, disable-able in 1K segments extended addressing.

MEM-16171A	*16K A & T*	**$149.95**

S-100 I/O Boards

SYSTEM SUPPORT 1 - CompuPro
Real time clock, three 16 bit interval timers, dual interrupt controllers(15 levels), up to 4K EPROM/RAM, RS-232C serial channel, provision for 9511A/9512 math chip.

IOX-1850A	*SS1 A & T*	**$359.95**
IOX-1850C	*SS1 CSC*	**$459.95**
IOX-1855A	*with 9511 A & T*	**$554.95**
IOX-1855C	*with 9511 CSC*	**$654.95**
IOX-1860A	*with 9512 A & T*	**$554.95**
IOX-1860C	*with 9512 CSC*	**$654.95**

INTERFACER 1 - CompuPro
2 serial I/O ports 50-19.2K baud.

IOI-1810A	*A & T*	**$218.95**
IOI-1810C	*CSC*	**$288.95**

INTERFACER 2 - CompuPro
3 parallel, 1 serial, & interrupt timer.

IOI-1820A	*A & T*	**$218.95**
IOI-1820C	*CSC*	**$288.95**

INTERFACER 3 - CompuPro
5 or 8 channel serial I/O board for interrupt driven multi-user systems up to 250K baud.

IOI-1835A	*5 port A & T*	**$558.95**
IOI-1835C	*5 port CSC*	**$628.95**
IOI-1838A	*8 port A & T*	**$628.95**
IOI-1838C	*8 port CSC*	**$749.95**

INTERFACER 4 - CompuPro
3 serial, 1 parallel, 1 Centronics parallel.

IOI-1840A	*A & T*	**$314.95**
IOI-1840C	*CSC*	**$414.95**

MPX - CompuPro
Multi-user I/O multiplexer & interrupt controller with on-board 8085A-2 CPU & 4K or 16K of RAM.

IOI-1875A	*4K MPX A & T*	**$444.95**
IOI-1875C	*4K MPX CSC*	**$534.95**
IOI-1880A	*16K MPX A & T*	**$584.95**
IOI-1880C	*16K MPX CSC*	**$674.95**

I/O-8 - SSM Microcomputer
Eight software programmable serial I/O ports, 110 -19.2K Baud, ideal for multi-user systems

IOI-1018A	*A & T*	**$469.95**

I/O-5 - SSM Microcomputer
Two serial & 3 parallel I/O ports, 110-19.2K Baud

IOI-1015A	*A & T*	**$289.95**

MPC-4 - SD Systems
Intelligent 4-port serial I/O card, on-board Z-80A, 2K RAM, 4K PROM area, on-board firmware, fully buffered, vectored interrupts, four CTC channels, add to SD Board set for powerful multi-user system

IOI-1504A	*A & T w/software*	**$495.00**

I/O-4 - SSM Microcomputer
2 serial I/O ports plus 2 parallel I/O ports.

IOI-1010B	*Bare board w/manual*	**$35.00**
IOI-1010K	*Kit with manual*	**$179.95**
IOI-1010A	*A & T with manual*	**$249.95**

2830 6 PORT SERIAL - C.C.S.
Six asynchronous RS-232C serial I/O ports with programmable baud rates.

IOI-1040A	*A & T with manual*	**$529.95**

2710 4 PORT SERIAL - C.C.S.
Four RS-232C serial I/O ports with full handshaking.

IOI-1060A	*A & T with manual*	**$319.95**

2719 2 SER & 2 PAR - C.C.S.
Two RS-232C serial I/O ports plus two 8 bit parallel I/O ports.

IOI-1080A	*A & T with manual*	**$349.95**

CIRCLE 9 ON READER SERVICE CARD

Prices may be slightly higher at our retail locations. Please call the store nearest you for local price and availability.

CP/M-86 Versus MS-DOS: A User's Perspective

by Steve Leibson

Editor's Introduction

MS-DOS was originally introduced in mid-1980 by Seattle Computer Products for their 8086 S-100-based microcomputer system. CP/M-86 was introduced in January 1981 by Digital Research, Inc. The July/August 1981 issue of *Microsystems* carried reviews of these two systems.

In the meantime, MS-DOS and CP/M-86 have been introduced for the IBM-PC, and both will soon be offered by vendors who are bringing new 8086/8088-based systems to market. Further, Lifeboat Associates have disclosed that they plan to offer SB-86, a version of MS-DOS which can be implemented by a knowledgeable user for his own particular hardware configuration. Hence, it appears that MS-DOS will be a "universal" system in much the same way that CP/M has already become one.

It is with these considerations in mind that we have decided to publish the following two articles which compare CP/M-86 and MS-DOS on two levels: the user interface level, and the programmer's level.

One last note: IBM has an exclusive license from Digital Research, Inc., to distribute CP/M-86 for the IBM-PC. It is expected that Digital Research, Inc., will themselves distribute Concurrent CP/M-86 for the IBM-PC to be available by the end of 1982. —*Sol Libes*

Ladieeeeees and Gentlemen! Tonight we have a special bout between the champ, Digital Research, and the challenger, Microsoft. The fight will determine the Welterweight (16-bit, low-end) World Champion Operating System for the 8086 and 8088 microprocessors. This article is a blow-by-blow comparison between Digital Research's version of CP/M for the 8086/8088 and Microsoft's operating system, MS-DOS. It is an even match—however, in this fight there could be a surprise winner.

What is an operating system?

Before we examine CP/M-86 and MS-DOS feature by feature, let's take a moment to consider what an operating system is and what it does for you and your computer. Figure 1 is a conceptual diagram of a total computer system. You, the user, interact with files, application programs, and the computer hardware to perform some task. The operating system is the glue that links all these components together.

When computer technology was still rather raw, each application program was responsible for the user, file and hardware interfaces. Operating systems were then introduced in order to standardize these interfaces so that a single application program would be able to run on several different computers. Thus an operating system offers some sort of standard method of interacting with users, files, and application programs.

Since every model of computer is different, something must change as the operating system is adapted to these computers. What changes is the interface to the hardware. That is why the arrows between the operating system and hardware portions of Figure 1 are labeled "implementation dependent." The hardware interface has to be specially coded for each model of computer; however, that task is much simpler than writing a new operating system for each computer.

Generally, an operating system alone is not a very useful piece of software. In addition to the application programs that make the computer do something useful, several utility programs are needed. These programs allow you to manage the files and resources that the operating system controls. Programs for copying files, formatting disks, and editing text are typical of utility programs supplied with CP/M-86 and MS-DOS.

In this article, I will discuss both the standardized features of CP/M-86 and MS-DOS and those features that are implementation dependent. The particular implementations I will be talking about are for the IBM-PC. The latest retail prices for these versions are $40 for MS-DOS and $240 for CP/M-86. They are both sold under the IBM label at all IBM-PC retail outlets.

A micro OS history

Both CP/M-86 and MS-DOS have their roots in the original CP/M operating system written by Dr. Gary Kildall for 8080-based systems. Digital Research, Inc., however, were slow in bringing out their 8086 version of CP/M. Seattle Computer Products, who were producing an 8086-based S-100 system, needed a DOS. They hired Tim Patterson to write it, and in mid-1980 they released SCP DOS-86. It looked, smelled, and tasted like CP/M, but the internal structure was different. Also, some of the user interfaces were changed to make the whole package easier to use. SPC also adapted Microsoft Basic (MBasic) to run under SPC DOS-86.

Steve Leibson, 4040 Greenbriar Blvd., Boulder, CO 80803

Microsoft bought the rights to the SCP DOS-86 and renamed it MS-DOS. Naturally, Microsoft adapted their highly successful Basic to their new operating system. IBM purchased the rights to MS-DOS and MBasic for their PC. IBM simply calls the operating system DOS. To simplify this article greatly, SCP DOS-86, MS-DOS, and DOS will all be referred to as MS-DOS.

Meanwhile, Digital Research was not going to let opportunity get away from them. They adapted CP/M to the 8086/8088 and called it CP/M-86, while also giving CP/M the new name CP/M-80. This was supposed to avoid any confusion about compatibility between CP/M programs and CP/M-86, but the attempt failed. Many people are still under the impression that programs that have been written to run on the 8080 microprocessor under CP/M will also run on an 8088 or 8086 system under CP/M-86. This is simply not so in the case of object code programs. There are several translator programs available to translate 8080 and Z80 source code to 8086 source code; however, these often require some additional manual translation as well.

Today we find a most interesting situation. Digital Research has the CP/M-86 operating system. Through the acquisition of Compiler Systems, Digital Research also owns CBasic, one of the dialects of Basic which has become popular with CP/M. Microsoft has its Basic and MS-DOS. I think it highly unlikely that Digital Research would endorse MS-DOS by adapting CBasic to that operating system.

It is equally unlikely that Microsoft will endorse CP/M-86 by adapting Microsoft Basic to Digital Research's operating system. Thus some of CP/M's universal charm has already been diluted in the 16-bit world. Microsoft and Digital Research, old-time allies, have become rivals.

[***Editor's Note:*** *Microsoft Basic has been converted by other vendors to run under CP/M-86. For example, TecMar (see the July/August 1981 issue of* Microsystems*) furnishes MBasic with their 8086-based S-100 system which supports CP/M-86. Also, Sorcim has available Pascal/M-86.*]

The user interface

An operator or user interacts with the computer directly through the operating system. In order to start a program or maintain files and other system resources, the user invokes operating system commands and runs the operating system utility programs. In the CP/M and and MS-DOS systems, commands and utility programs are initiated in

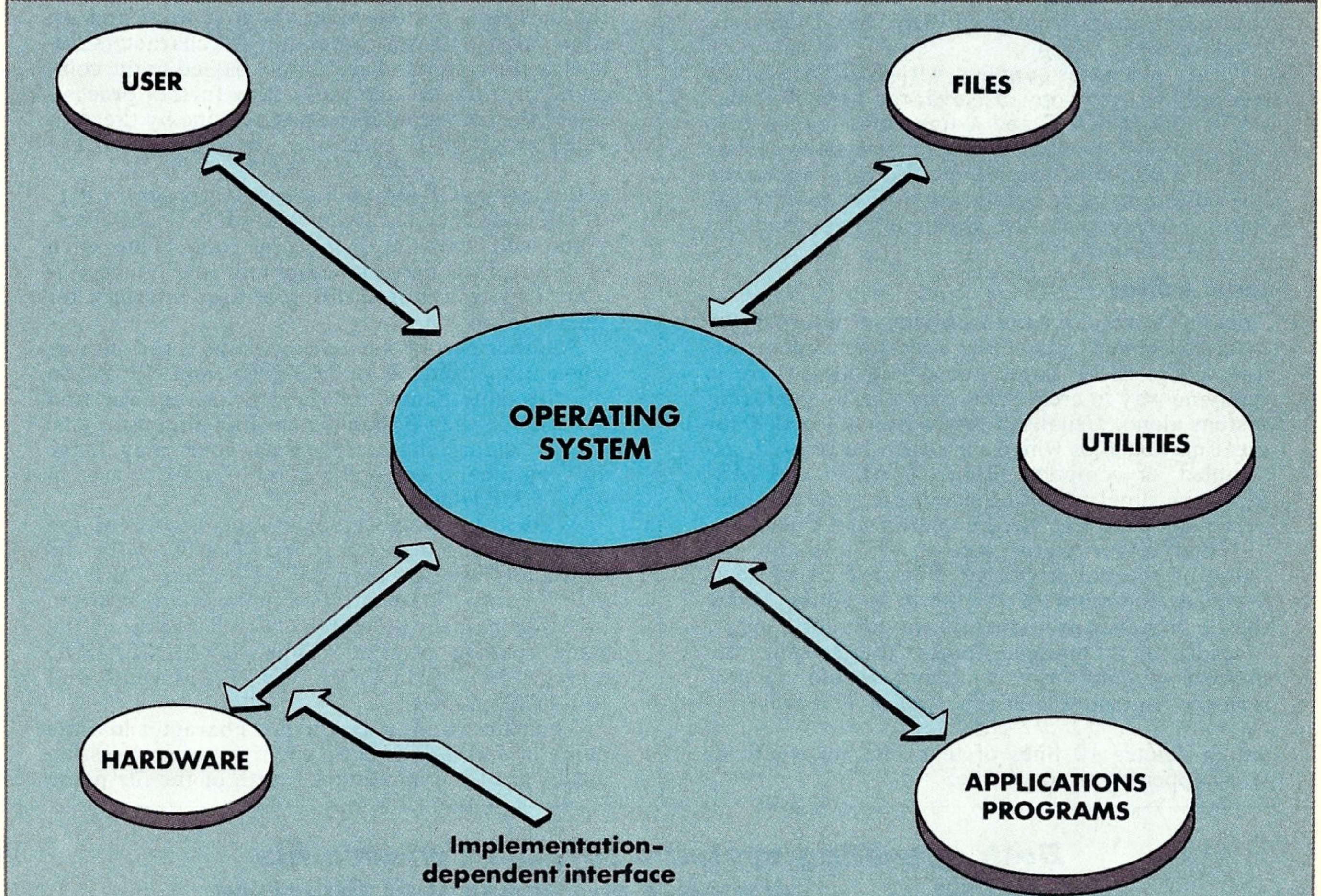


Figure 1.
Operating system—conceptual diagram.

the same manner. The name of the command or program is typed, followed by any parameters and a carriage return.

Commands that are built into the CP/M-86 operating system are called resident commands, while the commands built into MS-DOS are called internals. CP/M-86's resident commands are not the same as MS-DOS's internals, although there is quite a bit of overlap. Commands implemented with separate utility programs are called transients in CP/M-86 and external commands in MS-DOS. Table 1 is a comparison chart showing the major CP/M-86 commands versus MS-DOS commands.

You should see great similarity between the MS-DOS and CP/M-86 commands, an indication of their common ancestry. Both operating systems have many special transient or external commands that have been added especially for the IBM-PC, such as MODE in MS-DOS and NEWDISK, PROTOCOL, and TOD in CP/M-86.

One very nice transient command supplied with CP/M-86 is HELP. This is a program that will print out short instructions on the use of other CP/M-86 commands. It is like having a manual in the computer, accessible at any time. The closest equivalent to this supplied with MS-DOS is a quick reference card.

Other programs supplied with MS-DOS are a specially written copy of Advanced Microsoft Basic for the IBM-PC and a line editor called EDLIN. CP/M-86 also includes a line editor (ED) and an 8086/8088 assembler. We will look at the line editors in more detail, but a discussion of the Basic interpreter or the assembler is beyond the scope of this article.

Line editors

Though the resident and transient (or internal and external) commands of the operating systems allow you to move, copy, and delete files, there is really no way to create new files with an operating system alone. Usually a text editor is needed for creating text files which may then be printed, assembled, or compiled. Both CP/M-86 and MS-DOS are supplied with line editors and both look very much like ED, the line editor of CP/M.

Using either editor consists of invoking the name of the editor (either ED or EDLIN) followed by the name of the file to be edited. After that, commands are issued in the form of a letter, preceded by an optional number that specifies the number of times the command should be performed. An example of this type of syntax is:

10D

which deletes 10 lines of text starting wherever you happen to be in the file.

Any use of a line editor will quickly leave you with the feeling that there must be a better way. There is—it is called a screen editor. ED and its descendents have created a large opportunity in the market for superior editing capability. Many companies have introduced software products ranging from simple screen-oriented text editors to full word-processing systems.

A screen editor is what most computer owners buy immediately after purchasing their system. ED was written back when teleprinters were used for working with microcomputer systems. Today, with advanced terminals that have cursor positioning commands and memory-mapped screen hardware, there is little reason to be burdened with such an archaic utility.

The user interface

A computer user's impression of how easy or difficult it is to use a particular computer system is almost entirely determined by the user interface of the operating system. File maintenance, disk formatting, and application program initiation are all done through the operating system.

Both CP/M-86 and MS-DOS have identical user interfaces. Once a command line is typed in, the operating system takes the first word and assumes that that is a command. All characters following the command are simply passed to the command or program requested. Any further processing of the command line must be done by the activated program. For instance, the command:

PIP A:=B:*.*

activates the CP/M-86 file-copying program PIP, which takes the characters "A:=B:*.*" and interprets them with its own set of rules. Thus each program must perform command line interpretation. This results in a different user interface for each application.

Another part of the user interface is the naming convention used. Both MS-DOS and CP/M-86 use the same names for the disk drives. The first drive is A:, then B:, and so on. File names may be up to eight characters long and may have an optional three-character suffix, as in MYTEXT.DOC.

Wildcards are special characters used to match any single character or string. The two wildcard characters used by both CP/M-86 and MS-DOS are "?" and "*". A question mark in any position in a file name will match any character in the same relative position. Thus MYTEXT?.DOC matches MYTEXT.DOC, MYTEXT1.DOC, and MYTEXTZ.DOC.

The asterisk must be the last character in a file name or suffix. It is equivalent to typing question marks until the maximum length of the file name

Both operating systems support dynamic file allocation . . . minimum and maximum file sizes are 1K to 8M for CP/M-86, 1 byte to 1023M for MS-DOS

Move up to Microsystems

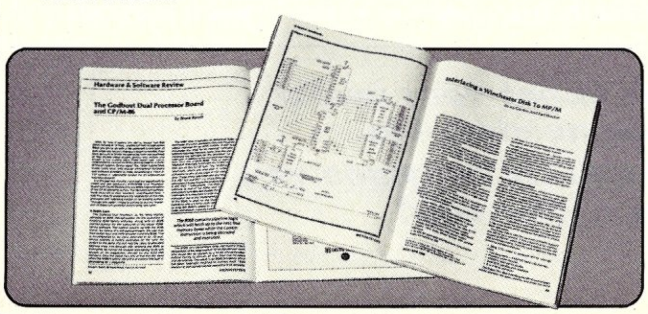
THE JOURNAL FOR SOPHISTICATED MICROCOMPUTER USERS.

Microsystems is not, in any sense, for beginners. Unlike "personal computer" magazines, *Microsystems* is written *exclusively* for expert users and designers in the software and support hardware field.

If you need a reliable, high-level, up-to-date journal devoted to CP/M,® MP/M,® MS-DOS,™ UNIX,® XENIX,™ OASIS and other operating systems, then *Microsystems* is for you. We feature tutorials on the modification and special use of systems, offer short-cut routines, show you debugging procedures, techniques for interfacing, and much more. Our one purpose is to keep the advanced microcomputer professional at the leading edge of DOS development and use.

In a few recent issues of Microsystems you would have seen:

- Reviews of four C compilers: BDS C, Small C, tiny c TWO™ and Whitesmiths C.
- A programming style comparison: Digital Research PL/I-80™ vs. Microsoft BASIC.
- Twenty-seven 16-Bit DOS compared.
- Virtual segment procedures under UCSD Pascal.
- An in-depth look at MP/M-80 II.™
- A simple 6-byte hexadecimal ASCII conversion routine.
- Using CP/M's undocumented "Autoload" feature.
- An explanation of double density disk controllers.
- A comparison of five popular S-100 disk controller cards.
- OS-1—a UNIX-like DOS with a CP/M adapter.

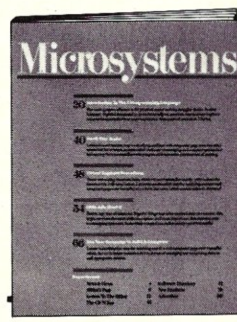


Every issue of *Microsystems* keeps you at the forefront of the industry in customizing software and adapting hardware to meet new requirements and standards. You'll also find industry news, book and new product reviews, plus our regular *Software Directory* listing newly available programs.

To get the high-level systems information you want, join the club of serious microcomputer professionals who subscribe to *Microsystems.* Use the coupon below to order, and save up to 33%!

PROFESSIONALS PLEASE NOTE:

Your subscription to Microsystems may be tax deductible. Check with your accountant.



Trademarks:
XENIX: Microsoft; tiny c TWO: tiny c assoc.; PL/I-80, MS-DOS, MP/M-80 II: Digital Research.
Registered Trademarks:
CP/M, MP/M: Digital Research; UNIX: Bell Laboratories.

Microsystems CN 1987
Morristown, New Jersey 07960

YES, enter my subscription to *Microsystems* for:

☐ 12 issues at $19.97—I save 20%.

☐ 24 issues at $36.97—I save 26%.

☐ 36 issues at $49.97—I save 33%.

Savings based on full 12-issue subscription price of $24.97.

Mr.
Ms. ______________________ 8H10
(please print full name)

Company ______________________

Address ______________________

City ______________________

State ____________ Zip ____________

CHECK ONE: ☐ Payment enclosed. ☐ Bill me later.

Offer valid in U.S. and possessions only. Please allow 30 to 60 days for delivery of first issue. Satisfaction guaranteed or a prompt refund for unmailed issues.

NEW SUBSCRIBERS ONLY

is reached. An example is MY*.DOC, which matches MYTEXT.DOC and MYTH.DOC, but not MYTEXT.LTR, because the asterisk matching is not carried over to the suffix. However, an asterisk placed after the period may be used to match the suffixes, also.

One interesting difference between the user interfaces of CP/M-86 and MS-DOS utilities is the order of specifying source and destination. In the above example from CP/M-86, "A:=B:*.*" specifies the destination A: first and then the source B:. For long-time users of CP/M, MS-DOS can become really frustrating. A similar command in MS-DOS would be:

COPY B:*.* A:

This command has the same effect, but the source and destination specifiers are typed in reverse order. You need to be really on your toes if you frequently switch between CP/M and MS-DOS.

A useful feature of both operating systems is the ability to perform batch processing. Back in the days before computers were personal, batch processing was the only type of computer interface available. Your job was performed in a batch. At the beginning of your card deck (remember those?) you placed some Job Control Cards that told the computer what to do with the rest of the cards. The whole job then ran without user intervention.

Batch processing

Batch processing for MS-DOS and CP/M-86 works in a similar manner, although each operating system has its own approach. MS-DOS has a special suffix for batch files called .BAT. Batch files are text files, but MS-DOS treats the characters in the file as if you had typed them in at the keyboard. Batch processing is initiated simply by typing in the name of the batch file, without the suffix.

In addition, a special batch file called AUTOEXEC.BAT will automatically be run whenever the computer is powered up. This is a very convenient method for configuring the system when first turned on and can be used to make turnkey systems that power up and immediately start running an application program.

Parameters may be passed to a batch file by typing them in on the same line as the name of the batch file. They are passed to the batch file as variables that occur in the file as a % followed by a number. The name of the batch file is passed in %0, with all following parameters passed successively as %1, %2, etc.

MS-DOS includes a PAUSE command that causes batch processing to halt until a key is pressed. This allows the operator to interact with the batch processing, to change disks for example. With this type of batch processing, a special batch file can be used to guide an untrained or inexperienced operator through a series of complex steps.

CP/M-86 also has batch capability. Again, a text file is prepared and the characters in the file are interpreted as keystrokes. However, CP/M-86 does not treat the batch file as a command. Instead it has the command SUBMIT, which takes the text file as a parameter. The text file must be of type .SUB. There are limitations on .SUB files. A line of text may not exceed 125 characters, and there is a limit of 128 lines. The whole file may not exceed 2048 characters.

Parameter passing is possible with CP/M-86's SUBMIT. The parameters start with $ and have sequential numbers following. The $0 parameter is the name of the .SUB file being SUBMITted. Any parameters typed after that on the command line take on sequential numbers, starting $1, $2, etc.

Files

A major function of an operating system is the maintenance and housekeeping associated with files. CP/M revolutionized microcomputer operating systems with dynamic file allocation. The human operator of the computer no longer needed to be concerned with how large a file was and whether the current task running on the system would exceed that size.

Dynamic file allocation takes care of those details. Both CP/M-86 and MS-DOS support dynamic file allocation. The two operating systems differ in the maximum and minimum possible size of a file. CP/M-86 allocates files in minimum blocks of 1024 bytes. That means the minimum file size for CP/M-86 is 1K. MS-DOS allocates files in minimum blocks of one sector. That means the minimum MS-DOS file is potentially 128 bytes although, practically, sectors are 256 or 512 bytes these days. The IBM-PC has 512 bytes per sector.

CP/M-86 uses the file allocation techniques that are found in CP/M. As a result, CP/M-86 has the same maximum file size limit of 8 megabytes. For floppy-based systems this is not a very big problem, but for hard disk systems that size may be smaller than desired. MS-DOS does not use CP/M-style file allocation, and files may be up to 1024 megabytes, which is considerably better for large systems. For most systems however, 8 megabytes is more than sufficient for the maximum file size.

MS-DOS's superior file size characteristics exact a price. More room is required in RAM to store the descriptions of where the pieces of the file

An operating system alone is not very useful . . . utility programs are needed, too. Any use of a line editor will quickly convince you that there must be a better way!

are actually stored. In operating systems that support dynamic file allocation, the files are rarely stored in one large piece. MS-DOS may require up to 6K bytes of RAM per storage device to store file information, while CP/M-86 requires only 1K. This extra RAM could have been a serious drawback for 8-bit computers with 64K RAM limitations, but on the 16-bit processors the extra RAM is not so readily missed.

One advantage that CP/M-86 does have is the ability to partition a storage device into user areas. CP/M and CP/M-86 support the concept of multiple users, although only one user can be active at a time. This is not multiuser capability, but it allows you to group your programs together logically. When you print out a directory, you won't be staring at an unmanageably long list of files, because each user has a private directory.

Files under MS-DOS are stamped with the time and the date. This assumes that either you enter the time and date when you power up your computer or your hardware has some way of obtaining this information automatically on power up. Several clock cards have appeared for the IBM-PC to support this feature. CP/M-86 does not support this feature, but Digital Research has been advertising that Concurrent CP/M-86 will do so. The time and date stamp is very convenient for version control of program source files, and also comes in handy for writing letters.

The applications program interface

An operating system offers the applications programmer a standardized set of capabilities with a standard technique for accessing them. These are functions such as "Print a character to the screen," "Get a character from the keyboard," and "Get a file from disk." CP/M supplies these standard functions in the form of a CALL to one master routing location. The number of the desired function is passed in register C and a parameter is passed in the DE register pair. This convention is highly specific to the 8080 microprocessor because of the specific registers used.

CP/M supports 38 different service requests that include console and disk input and output. The 38 requests are numbered 1 through 37 and 40. Requests 38 and 39 are supported only by Digital Research's multitasking operating system for 8-bit computers, MP/M. CP/M-86 supports the 38 CP/M requests and 10 of its own, numbered 50 through 59. These 10 new requests deal with memory management, which is much more complex on the 8088 than on the 8080.

A better operating system calling technique, made possible by the more advanced design of 16-bit processors (including the 8088 and 8086) is the trap. A trap is a software interrupt. With a single instruction which is usually shorter than a CALL, the program can call a subroutine through a special TRAP location.

Trap locations are in low-addressed memory, but each trap location contains the actual address of the routine being called, which can be anywhere in memory. The 8086/8088 supports 256 trap vectors that are called by the INT instruction. Calls in CP/M-86 are made using an INT 224 instruction instead of a CALL.

CP/M-86 supports the same system calls as CP/M, but the technique for making the requests differs slightly. The type of request is placed in the BL register, and any parameter is placed in the DX register. A single routing routine is again used to send the request to the proper place in the operating system.

MS-DOS supports three types of operating system requests. First, there are eight trap vectors that can be called directly. One of these is used to

Table 1. Major CP/M-86 and MS-DOS Commands

Function	MS-DOS Command	CP/M-86 Command
Batch processing	.BAT (*Note 1*)	SUBMIT, XSUB
Change a file name	RENAME *(R)*	REN *(R)*
Copy files	COPY *(R)*	PIP *(T)*
Make a file from RAM	—	SAVE *(R)*
File directory	DIR *(R)*	DIR *(R)*
Print a file to screen	TYPE *(R)*	TYPE *(R)*
Detailed file and disk info	CHKDSK *(T)*	STAT *(T)*
Disk duplication	DISKCOPY *(T)*	COPYDISK *(T)*
Delete a file	ERASE *(R)*	ERA *(R)*
Initialize a diskette	FORMAT *(T)*	NEWDISK *(T)*
Set time and date	TIME and DATE *(T)*	TOD *(T)*
Pause for keyboard (used for batch)	PAUSE *(R)*	—
File comparison	COMP *(T)*	—
Debug utility	DEBUG *(T)*	DDT-86 *(T)*
Assemble a program	*Note 2*	ASM-86 *(T)*
Linker	LINK *(T)*	*Note 3*
I/O redirection	MODE *(T)*	STAT *(T)*

(R) means resident or internal; (T) means transient or external.
Note 1. *Any text file with the suffix .BAT may be used as a Batch command file.*
Note 2. *MS-DOS's assembler is sold separately for $100.*
Note 3. *Linkers for CP/M-86 are sold with other software packages.*

make calls to other operating system functions beyond these eight. The other trap vectors are for program termination, the Control-Break keystroke exit, error handling, absolute disk I/O, and a special vector that allows a program to terminate but stay resident in the computer.

For all other functions, the second type is used: place the function request in the AH register and other parameters in various other registers as required by the individual calls. Then execute an INT 21 which actually causes the routine to be called.

The third type is for programs that were originally written for CP/M. The function number is placed in the CL register and a CALL is made to location 5 of the program's memory space. The other registers need to be set to appropriate values also. This second technique is valid only for MS-DOS functions 0 through 24 (hex). MS-DOS supports a total of 2E (hex) calls; they are similar to those supported by CP/M-86.

One of the functions supported by MS-DOS is quite interesting. It allows a program to terminate itself but still be resident in memory. Normally, when a program terminates, the memory allocated to it is reclaimed by the operating system. This feature can be used to advantage by interrupt-driven programs. You can load a program that first takes over an interrupt and then puts itself to sleep until its associated interrupt occurs.

Other useful functions available with MS-DOS are traps that allow a program to take over processing of exceptions. CP/M always intercepts Control-C and halts whatever is running. MS-DOS allows the program to take over processing of Control-C keystrokes. Another function can be used to prevent the return to the operating system for any reason. Software publishers will like this feature since it prevents a program from being loaded in, stopped, and analyzed. Other conditions that may be trapped are errors such as disk errors. An application program may substitute its own error handling for that of MS-DOS.

Program models

Since operating systems provide environments in which programs run, there must be some model of what a program looks like in memory. CP/M loads a program starting at location 100 Hex. Memory locations below 100 Hex are reserved for the operating system and contain buffer areas and jump tables. This sort of model works well for the 8080 because it has an absolute program counter. When the 8080 program counter contains the number 0, that means location 0 in memory.

This simple relationship is not true for the 8086/8088. Those CPUs have an instruction pointer that is added to the Code Segment register to obtain the actual memory location being referred to. In addition the 8088 has Data Segment, Stack Segment, and Extra Segment registers, which may all point to different parts of memory. Only if all of these segment registers contain zero do 8088 memory accesses look like those of the 8080. The program models of both MS-DOS and CP/M-86 reflect this difference.

MS-DOS supports two program models: EXE and COM. The COM files work in much the same way as CP/M programs. They are loaded and run from location 100 Hex according to the instruction pointer. All four of the segment registers are set to the beginning of the memory space allocated to the program by the operating system.

EXE type files are loaded with the Data Segment and Extra Segment registers set to the beginning of allocated memory, but the Code Segment, Stack Segment, Stack Pointer, and Instruction Pointer registers are set to values passed to the loader by the linker. That means the programmer can initialize these registers to whatever values are needed. EXE files are more complex to load and thus take more time in the loading process. The MS-DOS assembler produces EXE files, but a utility called EXE2BIN is supplied to convert EXE files into COM files.

CP/M-86 supports three program models: 8080, Small, and Compact. The 8080 model matches CP/M's model. All segment registers are set to the beginning of allocated memory, and the program is loaded at 100 Hex relative to that location. Small programs have separate code and data areas, while Compact programs support different values for each of the segment registers. The type of program model is selected by declarations in the assembler supplied with CP/M-86. The linker makes sure that the proper loading information is included in the loadable object file to inform the operating system as to the type of program in the file.

Development support

Operating systems alone aren't worth much. The true value of a computer is realized in its applications software. This must be written in assembly language (or machine code), or in a high-level language such as Basic, Fortran or Pascal.

IBM is marketing several software development packages for MS-DOS. These include Microsoft Basic (both interpreted and compiled), Pascal, and Fortran. A macro assembler has already been mentioned. The MBasic is a much enhanced version that supports the graphics and sound capabilities of the IBM-PC. The Pascal is also very powerful, with truly useful language extensions.

The independent software houses are not standing idle . . . Almost any programming language you might want to use is either already available or is about to become available for both systems.

CP/M-86 includes an assembler. Since Digital Research has Basic (both interpreted and compiled), Pascal, and PL/I in its stable of 8080 languages, it is probably safe to assume that 8086-compatible versions are forthcoming.

The independent software houses are not standing idle either. Versions of languages such as Forth, C, and Ada are being announced daily. Thus from a development standpoint almost any language you might want to program in is either already available or is about to become available for both operating systems.

The hardware interface

I have saved discussion of the hardware interface until last since it is highly implementation dependent. My version of the operating systems for the IBM-PC will have features not supported by all versions. I include a description of these implementations because each has some very good features.

Neither CP/M-86 nor MS-DOS can be SYSGENed on the IBM-PC. SYSGEN is a program that allows you to build an operating system tailored to your hardware. CP/M-86 is usually supplied with a SYSGEN program, but for the IBM-PC, both systems come pretailored. On power-up, the operating systems check internal switch settings on the IBM-PC motherboard and determine how much RAM, how many disk drives, and what type of display a particular system has attached. Both operating systems support up to 544K of RAM.

Automatic configuration is a double-edged sword. It is extremely convenient to have so much intelligence in the operating system that the novice user need never know that SYSGEN ever existed. It was never an easy program to use. On the other hand, the experienced computer user has his or her hands tied. No information is given, for either CP/M-86 or MS-DOS, as to how to add an I/O driver for a special piece of hardware.

[***Editor's Note:*** *CP/M, as furnished by Digital Research and others for systems other than the IBM-PC, does include the necessary utilities and documentation to perform a SYSGEN. In fact, the July/August 1981 issue of* Microsystems *carried an article which described the CP/M-86 SYSGEN technique in great detail and included a complete sample BIOS based on the Godbout 8085/8088 dual processor CPU card.*]

Pretailoring reduces IBM's support problems at the cost of increasing the frustration of many programmers. Eventually, someone will disassemble these operating systems and publish commented listings. It happened with the Apple computer, and it is bound to happen for the IBM as well.

Both operating systems support either single- or double-sided floppy disks on the IBM. Single-sided disks have 160K capacities, and double-sided disks have 320K. The double-sided disks are important for any substantial software support. Hard disks are not supported. [***Editor's Note:*** *CP/M-86 for non-IBM PC systems does not have these limitations.*]

MS-DOS has a very primitive implementation for the IBM screen. It is simply a glass printer. Characters are sequentially sent to the screen and backspace, carriage return and line feed are supported. CP/M-86 has a much better set of screen controls. Escape sequences are implemented to clear the screen, clear to the end of the current line, and position the cursor. This makes the screen look more like an intelligent terminal, although the escape sequences picked do not match any standard terminal protocols. Neither operating system supports the IBM-PC's graphics capabilities.

Both operating systems do a very nice job of supporting the IBM serial port. CP/M-86 has a SPEED utility that allows you to program the serial port's bit rate, number of bits per character, and parity. Also supplied is the PROTOCOL utility that programs the serial port to support either of the common software handshakes: XON/XOFF and ETX/ACK. These protocols are extremely important for communicating with serially interfaced printers. The IBM-PC version of CP/M-86 does not support a hardware handshake for the serial port.

MS-DOS uses a single MODE utility to program the serial port. Baud rate, number of bits per character, parity, and number of stop bits may be programmed. A hardware handshake may also be specified. Software handshakes are not supported. Both operating systems support one or two serial ports.

Documentation

Manuals for both CP/M-86 and MS-DOS are very good compared to their predecessors. They are typeset and indexed. This is IBM's influence. The manuals are supplied in the small three-ring binders for which IBM-PC software is becoming famous.

What I don't like about the documentation is that it is incomplete from a programmer's standpoint. There is no documentation on I/O drivers. Custom systems may be a thing of the past. I can hear the marketeers shouting hallelujah. [***Editor's Note:*** *This is not true for the non-IBM-PC version of CP/M-86.*]

Some of CP/M's universal charm has already become diluted in the 16-bit world. Microsoft and Digital Research, old-time allies, have become rivals.

Personal comments

There is no clear winner here except IBM, which markets both operating systems for its computer. Performance levels on my floppy disk system are close enough to call it a draw. CP/M-86's file size limitations are irrelevant on the IBM-PC also.

I find that MS-DOS has an easier-to-use batch facility. The AUTOEXEC.BAT feature is especially important for adding patches to the operating system for those vendors who have disassembled MS-DOS sufficiently to add their own hardware.

As to support of the IBM hardware, again it is too close to call. CP/M-86 does a better job of screen control, but this is not so important in a memory-mapped system.

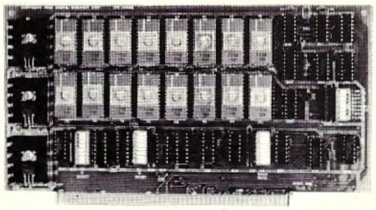
The real question is: Why pay six times as much for CP/M-86 when the two systems are so close in performance and capabilities? Every IBM-PC sold with floppies goes out with a copy of MS-DOS because it is so inexpensive. All of IBM's applications run under MS-DOS and not CP/M-86. The third-party vendors have also given the nod to MS-DOS, with programs such as WordStar and vendors such as Lifeboat Associates crossing over from CP/M. At least for the PC, MS-DOS will be the leader for quite some time.

CP/M, CP/M-80, and CP/M-86 are trademarks of Digital Research.
8080, 8088, and 8086 and Intel are trademarks of Intel Corporation.
Z80 is a trademark of Zilog, Inc.
SCP DOS-86 is a trademark of Seattle Computer Products.
MS-DOS is a trademark of Microsoft, Inc.
SB-86 is a trademark of Lifeboat Associates.

Steve Leibson is a Senior Engineer for the Cadnetics Corporation, a Boulder-based manufacturer of low-cost computer-aided design systems. He also has his own firm, Data Press, which has published *The Great Small Business Computer Ripoff*, a book for people in business considering their first computer purchase. Steve is also the editor of the *Interrupt* newsletter for the Denver Amateur Computer Society.

DIGITAL RESEARCH COMPUTERS
(214) 271-3538

32K S-100 EPROM CARD
PRICE CUT!



$59.95

USES 2716's

Blank PC Board - $34

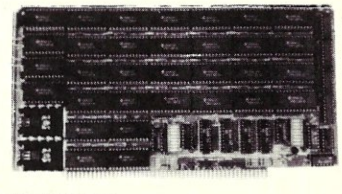
ASSEMBLED & TESTED ADD $30

SPECIAL: 2716 EPROM's (450 NS) Are $4.95 Ea. With Above Kit.

KIT FEATURES:
1. Uses +5V only 2716 (2Kx8) EPROM's.
2. Allows up to 32K of software on line!
3. IEEE S-100 Compatible.
4. Addressable as two independent 16K blocks.
5. Cromemco extended or Northstar bank select.
6. On board wait state circuitry if needed.
7. Any or all EPROM locations can be disabled.
8. Double sided PC board, solder-masked, silk-screened.
9. Gold plated contact fingers.
10. Unselected EPROM's automatically powered down for low power.
11. Fully buffered and bypassed.
12. Easy and quick to assemble.

64K S100 STATIC RAM

$299.00 KIT



NEW!

LOW POWER!

RAM OR EPROM!

BLANK PC BOARD WITH DOCUMENTATION $55

SUPPORT ICs + CAPS $17.50

FULL SOCKET SET $14.50

FULLY SUPPORTS THE NEW IEEE 696 S100 STANDARD (AS PROPOSED)

FOR 56K KIT $249

ASSEMBLED AND TESTED ADD $40

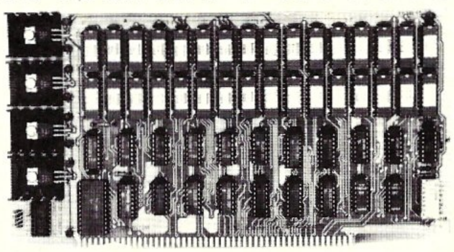
FEATURES:
* Uses new 2K x 8 (TMM 2016 or HM 6116) RAMs.
* Fully supports IEEE 696 24 BIT Extended Addressing.
* 64K draws only approximately 500 MA.
* 200 NS RAMs are standard. (TOSHIBA makes TMM 2016s as fast as 100 NS. FOR YOUR HIGH SPEED APPLICATIONS.)
* SUPPORTS PHANTOM (BOTH LOWER 32K AND ENTIRE BOARD).
* 2716 EPROMs may be installed in any of top 48K.
* Any of the top 8K (E000 H AND ABOVE) may be disabled to provide windows to eliminate any possible conflicts with your system monitor, disk controller, etc.
* Perfect for small systems since BOTH RAM and EPROM may co-exist on the same board.
* BOARD may be partially populated as 56K.

16K STATIC RAM KIT-S 100 BUSS

PRICE CUT!

$119.95 KIT

FOR 4MHZ ADD $10



KIT FEATURES:
1. Addressable as four separate 4K Blocks.
2. ON BOARD BANK SELECT circuitry (Cromemco Standard!). Allows up to 512K on line!
3. Uses 2114 (450NS) 4K Static Rams.
4. ON BOARD SELECTABLE WAIT STATES.
5. Double sided PC Board, with solder mask and silk screened layout. Gold plated contact fingers.
6. All address and data lines fully buffered.
7. Kit includes ALL parts and sockets.
8. PHANTOM is jumpered to PIN 67.
9. LOW POWER: under 1.5 amps TYPICAL from the +8 Volt Buss.
10. Blank PC Board can be populated as any multiple of 4K.

BLANK PC BOARD W/DATA-$33

LOW PROFILE SOCKET SET-$12

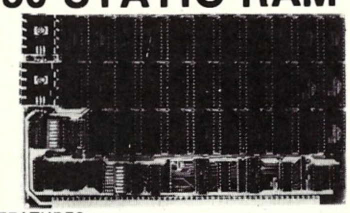
SUPPORT IC'S & CAPS-$19.95

ASSEMBLED & TESTED-ADD $35

OUR #1 SELLING RAM BOARD!

64K SS-50 STATIC RAM

$219.00 (48K KIT)



NEW!

LOW POWER!

RAM OR EPROM!

BLANK PC BOARD WITH DOCUMENTATION $52

SUPPORT ICs + CAPS $18.00

FULL SOCKET SET $15.00

56K Kit $269

64K Kit $319

ASSEMBLED AND TESTED ADD $40

FEATURES:
* Uses new 2K x 8 (TMM 2016 or HM 6116) RAMs.
* Fully supports Extended Addressing.
* 64K draws only approximately 500 MA.
* 200 NS RAMs are standard. (TOSHIBA makes TMM 2016s as fast as 100 NS. FOR YOUR HIGH SPEED APPLICATIONS.)
* Board is configured as 3-16K blocks and 8-2K blocks (within any 64K block) for maximum flexibility.
* 2716 EPROMs may be installed anywhere on Board.
* Top 16K may be disabled in 2K blocks to avoid any I/O conflicts.
* One Board supports both RAM and EPROM.
* RAM supports 2MHZ operation at no extra charge!
* Board may be partially populated in 16K increments.

STEREO! S-100 SOUND COMPUTER BOARD

NEW!

At last, an S-100 Board that unleashes the full power of two unbelievable General Instruments AY3-8910 NMOS computer sound IC's. Allows you under total computer control to generate an infinite number of special sound effects for games or any other program. Sounds can be called in BASIC, ASSEMBLY LANGUAGE, etc.

COMPLETE KIT! $69.95 (WITH DATA MANUAL)

BLANK PC BOARD W/DATA $31

KIT FEATURES:
* TWO GI SOUND COMPUTER IC'S.
* FOUR PARALLEL I/O PORTS ON BOARD.
* USES ON BOARD AUDIO AMPS OR YOUR STEREO.
* ON BOARD PROTO TYPING AREA.
* ALL SOCKETS, PARTS AND HARDWARE ARE INCLUDED.
* PC BOARD IS SOLDERMASKED, SILK SCREENED, WITH GOLD CONTACTS.
* EASY, QUICK, AND FUN TO BUILD. WITH FULL INSTRUCTIONS.
* USES PROGRAMMED I/O FOR MAXIMUM SYSTEM FLEXIBILITY.

Both Basic and Assembly Language Programming examples are included.

SOFTWARE:

SCL™ is now available! Our Sound Command Language makes writing Sound Effects programs a SNAP! SCL™ also includes routines for Register-Examine-Modify, Memory-Examine-Modify, and Play-Memory. SCL™ is available on CP/M* compatible diskette or 2708 or 2716. Diskette - **$24.95** 2708 - **$19.95** 2716 - **$29.95.** Diskette includes the source. EPROM'S are ORG at E000H. **(Diskette is 8 Inch Soft Sectored)**

NEW! G.I. COMPUTER SOUND CHIP

AY3-8910. As featured in July, 1979 BYTE! A fantastically powerful Sound & Music Generator. Perfect for use with any 8 Bit Microprocessor. Contains: 3 Tone Channels, Noise Generator, 3 Channels of Amplitude Control, 16 bit Envelope Period Control, 2-8 Bit Parallel I/O, 3 D to A Converters, plus much more! All in one 40 Pin DIP. Super easy interface to the S-100 or other busses.

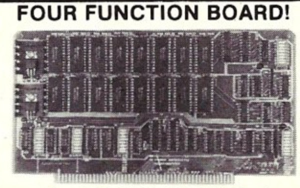
SPECIAL OFFER: ~~$14.95~~ **each** $9.95 **PRICE CUT!** Add $3 for 60 page Data Manual.

32K S100 EPROM/STATIC RAM

NEW! FOUR FUNCTION BOARD! **NEW!**

EPROM II FULL EPROM KIT $80.00

A&T EPROM ADD $35.00



BLANK PC BOARD WITH DATA $39.95

SUPPORT IC'S PLUS CAPS $23.00

FULL SOCKET SET $18

We took our very popular 32K S100 EPROM Card and added additional logic to create a more versatile EPROM/RAM Board.

FEATURES:
* This one board can be used in any one of four ways:
  - A. As a 32K 2716 EPROM Board
  - B. As a 32K 2732 EPROM Board (Using Every Other Socket)
  - C. As a mixed 32K 2716 EPROM/2K x 8 RAM Board
  - D. As a 32K Static RAM Board
* Uses New 2K x 8 (TMM2016 or HM6116) RAM's
* Fully Supports IEEE 696 Buss Standard (As Proposed)
* Supports 24 Bit Extended Adressing
* 200 NS (FAST!) RAM'S are standard on the RAM Kit
* Supports both Cromemco and North Star Bank Select
* Supports Phantom
* On Board wait State Generator
* Every 2K Block may be disabled
* Addressed as two separate 16K Blocks on any 64K Boundary
* Perfect for MP/M* Systems
* RAM Kit is very low power (300 MA typical)

32K STATIC RAM KIT — $169.95

For RAM Kit A&T - Add $40

Digital Research Computers
(OF TEXAS)
P.O. BOX 401565 • GARLAND, TEXAS 75040 • (214) 271-3538

TERMS: Add $2.00 postage. We pay balance. Orders under $15 add 75¢ handling. No C.O.D. We accept Visa and MasterCharge. Tex. Res. add 5% Tax. Foreign orders (except Canada) add 20% P & H. Orders over $50, add 85¢ for insurance.

ALL SALES ARE MADE SUBJECT TO THE TERMS OF OUR 90 DAY LIMITED WARRANTY. A COPY OF THIS WARRANTY IS AVAILABLE FREE, ON REQUEST.

*TRADEMARK OF DIGITAL RESEARCH.

WE ARE NOT ASSOCIATED WITH DIGITAL RESEARCH OF CALIFORNIA, THE SUPPLIERS OF CPM SOFTWARE.

CP/M-86 Versus MS-DOS: A Programmer's Perspective

by Neil Colvin

In this article I will attempt to provide some basic insight into the technical similarities and differences between Digital Research's CP/M-86 and Microsoft's MS-DOS. I will, for the most part, restrict myself to a functional comparison, with the issues of bits and bytes being better left to the system implementors.

First, however, some background. My company (Phoenix Software Associates, Ltd.) is a software house that specializes in "system" software. A large percentage of the software we have produced to date is assembler code, although we are rapidly moving to C for all new development. One part of our recent efforts has been the conversion of a number of 8080/Z80-based programs to run on the 8086/8088. This has involved not only the code conversion, but also the reintegration of the software with the new operating system environment. Because of our early entry into the 16-bit marketplace, we have been forced to integrate these converted programs, as well as our new software products, with both CP/M-86 and MS-DOS. Aside from the inconvience of supporting two operating system environments, this experience has provided us with valuable insights into the strengths and weaknesses of both systems. I hope that by sharing some of this insight, those of you faced with decisions as to which system to use, or support, will better understand some of the many technical issues affecting such a decision.

The major functional areas on which I am going to comment include: program management, program control, memory management, serial I/O, disk I/O, file management, command processing, standard support utilities, CP/M-80 conversion effort, and reliability. I should note that this comparison is based on the currently released versions of both operating systems, and that both Digital Research and Microsoft have announced future product upgrades which may address a number of the areas that I will discuss.

Program management

"Program management" refers to the types of program models supported by the operating system's "program loader." Both CP/M-86 and MS-DOS support multiple program models to provide flexible control of the 8086's program/memory architecture. CP/M-86 supports three models: 8080, SMALL, and COMPACT. MS-DOS supports two models: 8080 (COM) and 8086 (EXE). These models provide different initial mappings of the program code on disk to the 8086 memory, and different initial settings of the 8086 segment registers. Figure 1 shows the 8080 program model.

Both operating systems provide an 8080 program model, derived in part from the CP/M-80 program model. In these models, the 8086 code (CS), data (DS) and stack (ES) segments are one and the same, and code and data are intermixed in the segment. The code starts at 100H relative to the start of the segment, and low memory in the segment (0-FFH) is predefined in much the same way as in CP/M-80. I will discuss the use of low memory in greater detail when I compare CP/M-80 compatibility.

Both operating systems also provide a program model that supports separate code and data/stack segments (Figure 2). This allows for additional automatic memory allocation for these segments, as well as the development of sharable code segments for use in multitasking environments. MS-DOS, however, provides an additional feature in this program model (shown in Figure 3) that is not available under CP/M-86: program relocation.

The 8086 architecture restricts a single segment to 64K. Since one of the advantages of the 8086 is its larger address space, it is obviously desirable to allow programs that are larger than 64K to run in the system. Both operating systems will allow multiple segments of code to be included in the program, but only MS-DOS will automatically relocate the inner-segment linkages (in the form of in-line FAR JUMPs, FAR CALLs, and double-word pointers) at the time of program loading.

Under CP/M-86, in its third program model (Figure 3), the segment bases of these additional

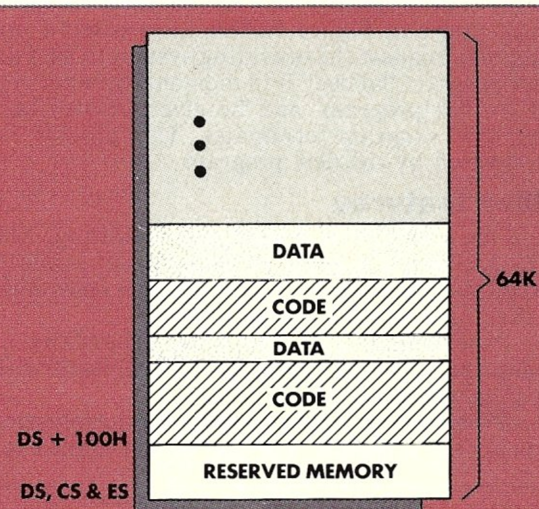


Figure 1—CP/M-80 Program Model

Neil Colvin, Phoenix Software Associates, Ltd., Box 207, N. Easton, MA 02356

program segments (AS) are available in the memory image, but must be explicitly managed by the program code to produce FAR CALL and FAR JUMP pointers.

Memory management

Closely related to the area of program management is that of memory management. I have already discussed the manner in which each operating system allocates memory when a program is loaded. By memory management, I am referring to the facilities within the operating system that are available for the program to use in obtaining additional memory resources during execution. The two operating systems differ greatly in their approaches to this problem.

MS-DOS uses an approach similar to CP/M-80, shown in Figure 4, for its memory management. After a program is loaded, certain locations in its low data area contain the addresses of the top of the current data segment, and the top of all available memory. The program has access to all memory from the base of its code segment through the top of memory, and may manage it as it desires. One consequence of this technique is that MS-DOS can support only machine environments that have contiguous user RAM.

CP/M-86, as shown in Figure 4, provides a set of system functions for allocating and releasing dynamic memory space. Memory may be allocated out of the free memory pool, or at absolute memory locations (providing control over access to memory-mapped devices or other nonsharable memory resources). Memory segments of specific sizes may be requested, or a request may be made for the maximum memory segment available. One consequence of this allocation scheme is that the available memory under CP/M-86 need not be contiguous, so that systems in which memory-mapped devices or ROMs intrude upon the memory space may be easily supported.

One additional feature of both operating systems is that programs may be "stacked" in memory as shown in Figure 5. This means that a program may request the operating system to load another program "above" it in memory, transfer control to that program, and be given control back when that program terminates. This process can be repeated by the new program.

Other features

Once the program is loaded and executing, MS-DOS provides a number of features not available in CP/M-86 for use by the program in controlling its execution process. These include:

—Programmable trap for CTL-C handling, allowing the program to handle CTL-C itself without program termination.

—Programmable trap for program termination, allowing the program to inhibit return to the operating system for any reason. This trap is stacked when a program loads a new program, so that the final program termination of the new program will always return to the calling program, even if the new program has set its own termination trap.

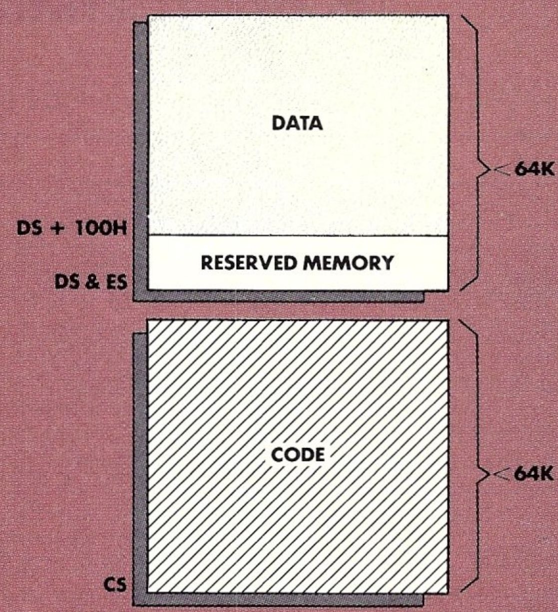


Figure 2—Separated Code and Data Space Models

—Programmable trap for critical error handling, allowing the program to directly handle disk I/O errors and other errors that would normally cause program termination. Detailed error codes and error recovery options are provided for use by the error routines.

—Memory lock option, allowing a program to remain memory-resident after termination. This allows a program to link itself into the interrupt vectors or device driver area, and then terminate without the memory area it is occupying being reused until a cold start of the system is done.

—Time and date support, with calls for reading and setting the system time, and internal support for a real-time clock.

Serial I/O

In comparing the serial I/O capabilities of the two operating systems, there are many little differences and one major one. I will discuss the minor differences first. CP/M-86 provides direct access to the BIOS functions for serial I/O through a system call, whereas MS-DOS does not provide any direct access to the BIOS level I/O. MS-DOS does offer an expanded set of system calls for dealing with console I/O however, including direct and indirect Read Without Echo and Flush Keyboard

A major difficulty with the current MS-DOS is the lack of any form of directory partitioning.

Buffer (for interrupt-driven keyboards). MS-DOS does not provide any equivalent to the CP/M-86 List Status call to determine if the list device is ready to receive a character (a feature used by many programs for background printing.

A second minor difference is in the I/O byte handling. Although both operating systems support four logical devices (console, list, reader, punch), CP/M-86 supports the use of an IOBYTE by the BIOS to provide dynamic serial device assignments of up to four physical devices per logical device, if desired, as well as the use of the STAT utility to change those assignments. MS-DOS provides no equivalent facility.

The third minor difference has to do with the sophistication of the "buffered console input" function. Both systems provide editing capability while inputting, but MS-DOS provides a user-programmable line editor with facilities similar to those found in the editor of Microsoft's Basic interpreter.

The major serial I/O difference, however, lies in MS-DOS's support of a "device-independent" I/O facility for serial devices. This allows the standard file I/O calls (i.e., read and write) to be done to special file names (CON, PRN, NULL, and AUX) to access the serial devices. This allows programs to be written that do not need to distinguish between serial I/O to disk files and doing serial I/O to other actual serial devices (in sort of a Unix fashion).

Disk I/O

Although the areas of disk I/O and file management are closely related, there are a few comparisons that can be made about the basic disk I/O approaches taken by the two systems. CP/M-86 follows precisely in CP/M-80's footsteps in approaching disk I/O. It assumes a logical sector size of 128 bytes, and leaves it to the system implementor to do the blocking and deblocking required to support that size. It also assumes one logical sector at a time for all I/O operations, with an implicit assumption that the implementor will do whatever interleaving or other hardware optimization is required.

MS-DOS takes a different approach. It assumes that the hardware-dependent disk drivers need only concern themselves with reading and writing sequential physical sectors. There are no arbitrary restrictions placed on physical sector size (it does not even need to be a power of two), and MS-DOS performs internally any blocking and deblocking necessary. In addition, MS-DOS requests multiple physical sector reads and writes from the driver whenever possible, allowing the use of noninterlaced disk structures for maximum speed.

These differences in approach to basic disk I/O are reflected in the file management strategies of the two operating systems.

File management

Before going into a detailed comparison of the file management capabilities of the two systems, here are some basic quantitative comparisons.

	CP/M-86	MS-DOS
Maximum allocation block size	16K	2048K
Minimum allocation block size	1K	1 sector

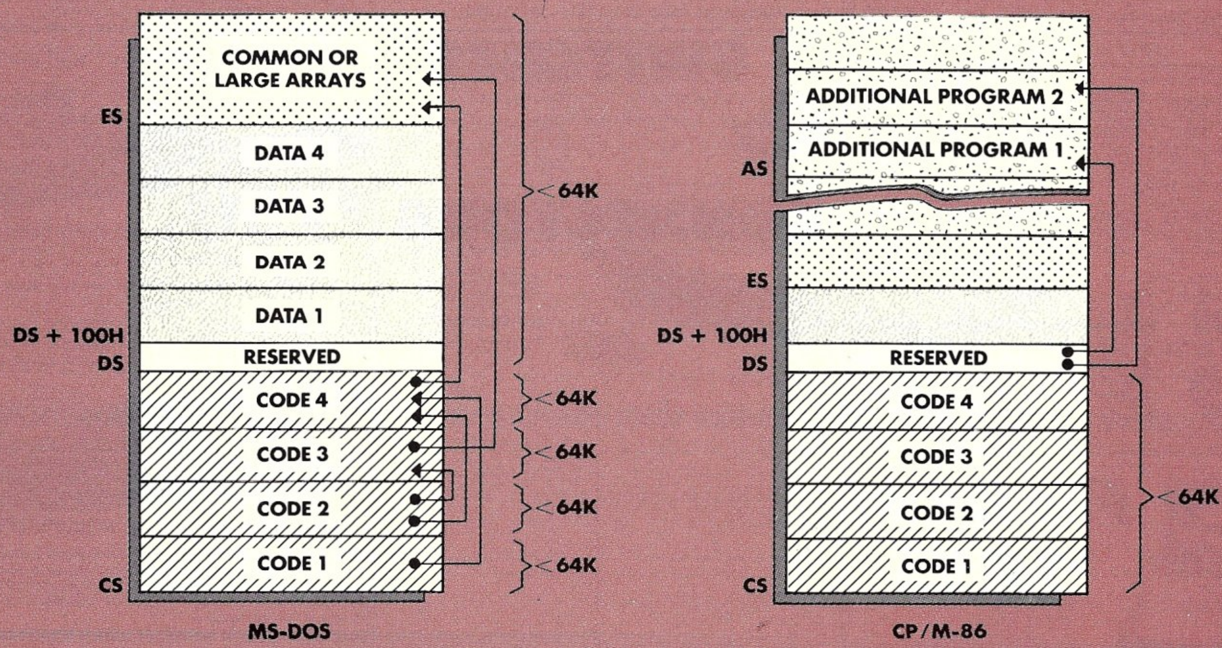


Figure 3—Program Models

California Digital

Post Office Box 3097M • Torrance, California 90503

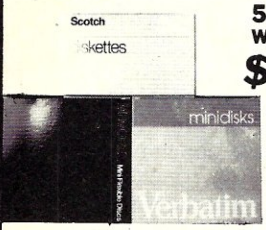
★ FREE ★

Plastic library case supplied with all diskettes purchased from California Digital

$24.95

Private labeled for California Digital by one of the most respected producers of magnetic media. Each diskette is certified double density at 40 tracks. To insure extended media life each diskette is manufactured with a reinforced hub-hole. And of course, a plastic library case is included with every box of diskettes. MMD-CD5(01)(10) Please specify computer or required sectors.

Ten boxes $22.75 One hundred boxes $21.50



5¼" DISKETTES WITH LIBRARY CASE

$26.50

Your Choice
SCOTCH
MEMOREX
VERBATIM

Single Side Double Density

	Soft Sector	10 Sector	16 Sector	
SCOTCH	744D-0	744D-10	744D-16	26.50
MEMOREX	3481	3483	3485	26.50
VERBATIM	525-01	525-10	NA	26.50
MAXELL	MD1	MH1-10	MH1-16	29.85
DYSAN	104/1D	107/1D	NA	45.00

Double Side Double Density

SCOTCH	745-0	745-10	745-16	42.50
VERBATIM	550-01	550-10	NA	42.50
MAXELL	MD2-D	MH2-10D	MH2-16D	45.00
DYSAN	104/2D	107/2D	NA	49.50
DYSAN 96	204/2D	NA	NA	59.50

EIGHT INCH DISKETTES

Single Side Single Density		
SCOTCH	740-0	29.50
MEMOREX	3060	29.50
DYSAN	3740/1	39.50
Thirty Two Sector		
SCOTCH	740-32	29.50

Single Side Double Density		
SCOTCH	741-0	39.00
MEMOREX	3090	35.00
DYSAN	3740/D	57.50
Double side Double Density		
SCOTCH	743-0	47.50
MEMOREX	3114	39.50
DYSAN	3740/2D	65.00

Scotch Head Cleaning Kits 5¼ & 8"	$24.95
Plastic Library Cases 5¼ & 8"	2.95
Diskette Flip Tubs fifty diskettes 5¼"	
Diskette Flip Tubs fifty diskettes 8"	

Microswitch ASCII KEYBOARD $79

Each keyboard contains 81 high reliability Hall Effect keys. Outputs seven bit parallel ASCII MIC-81SD5 3 Lbs.

Flip & File $18.95

Plastic storage file allows immediate access to over 50 diskettes.

5 1/4"	INC-525	$18.95	4 lbs.
8"	INC-800	$19.95	6 lbs.

MEMORY

2716 EPROM	64K DYNAMIC
4.50	6.95
450ns	4164 200ns
16K STATIC	**16K DYNAMIC**
7.00	1.95
6116 200ns	4116 200ns

2732 EPROM SALE $6.95

DYNAMIC MEMORY

	1-31	32+	100+
4116-150ns 16k	1.95	1.85	1.75
4116-200ns 16k	1.75	1.65	1.50
4164-150ns 64k Japan 128 refresh	8.95	8.25	7.90
4164-150ns 64k Texas Inst. 256 refresh	7.00		
4164-200ns 64k Japan 128 refresh	8.50	7.75	7.50
41256-200 256K dynamic memory	call for pricing		

STATIC MEMORY

21L02-200ns 1k	1.49	1.29	1.15
21L02-450ns 1k	1.29	1.15	.99
2112-450ns 2k	3.95	3.50	3.25
214-300ns 4k			
2114-450ns 4k			
4044-250ns 4k			
4044-450ns 4k			
5257-300ns 4k			
6116-P3 150ns 16k 24 pin	7.50	7.25	6.90
6116-P4 200ns 16k 24 pin	7.00	6.75	6.65
6167 / 2167 100ns 16k 20 pin	9.95	9.50	8.90

EPROMS

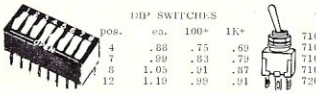
2708-450ns 8k	3.50	2.95	2.75
2716-450ns 16k	4.50	4.15	3.75
2716TMS-450ns 16k tri-voltage	7.95	7.65	7.25
2516-450ns 16k	6.95	5.95	5.45
2732-350ns 16k	8.50	8.00	7.60
2732-450ns 16k	6.95	6.65	6.35
2532-450ns 16k	10.50	9.90	9.50
2764-450ns 64k	14.95	13.75	12.50
27128-450ns 128k	call for pricing		

$49.95

ASTEC SWITCHING POWER SUPPLY

Output; +5v. 3A; +12v .9A; −12v .075A.
Suitable for Apple look-a-like. ATC-1298

SWITCHES



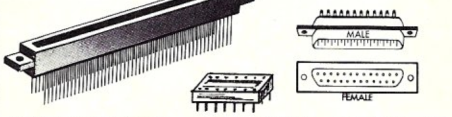
DIP SWITCHES

pos.	ea.	100+	1K+
4	.88	.75	.69
7	.99	.83	.79
8	1.05	.91	.87
12	1.19	.99	.91

TOGGLE

		ea.	100+
7101	on/on	1.19	.98
7103	on/off/on	1.39	1.19
7107	mnt. on	1.39	1.19
7108	mnt. on	1.39	1.19
7205	dpdt mnt	1.85	1.65

CONNECTORS



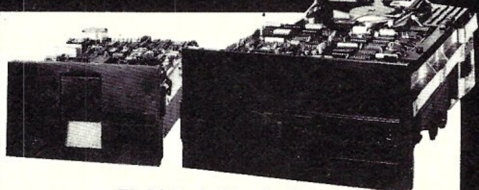
GOLD EDGE CONNECTORS

S-100 .125" centers	each	10+
Imsai solder .250" row	$2.95	$2.50
Imsai wire wrap (TI)	3.95	3.50
Sullins Hi-Rel. .250"	4.50	4.00
Sullins Hi-Rel. W/W	5.35	4.90
Sullins / Altair .140"	4.95	4.50
.156" Centers (standard)		
22/44 Kim Eyelet	2.50	2.15
36/72 Digital Group S/T	5.95	5.50
36/72 Digital Group W/W	6.60	6.15
43/86 Motorola 6800 S/T	6.60	6.15
43/86 Moto. 6800 W/W	7.00	6.85

INTEGRATED CIRCUIT SOCKETS

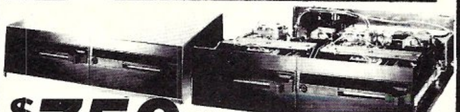
	Low Profile each	Low Profile 100+	Wire Wrap each	Wire Wrap 100+
8 pin	$.10	$.09	$.46	$.41
14 pin	.10	.09	.45	.41
16 pin	.12	.11	.50	.45
18 pin	.15	.13	.68	.61
24 pin	.26	.24	.94	.87
40 pin	.42	.40	1.60	1.47

"D" Type	each	10-24	25+
DE9P male	$1.60	$1.40	$1.30
DE9S female	2.25	2.00	1.90
DE hood	1.50	1.35	1.20
DA15P male	2.35	2.15	2.00
DA15S female	3.25	3.10	2.90
DA hood 2/P	1.60	1.35	1.30
DB25P male	2.50	2.35	2.25
DB25S female	3.35	3.15	3.05
DB hood 2/P	1.35	1.15	1.05
DC37P male	4.20	4.00	3.70
DC37S female	6.00	5.75	5.50
DC hood 2/P	2.25	2.00	1.75
DD50P male	5.50	5.10	4.75
DD50S female	9.40	8.60	8.00
DD50 hood 2/P	2.60	2.40	2.10
CENTRONICS			
57-30360	7.95	6.75	5.75
RIBBON CABLE CONNECTORS			
17/34 5" disk	4.85	4.15	3.95
20/40 TRS-80	5.65	5.05	4.70
25/50 8" disk	5.90	5.15	4.90



Eight Inch Single Sided	One	Two	Ten
SHUGART SA801R	$395	385	375
SIEMENS FDD100-8	259	259	225
TANDON 848-1 SLIMLINE	379	369	359
Eight Inch Double Sided			
SHUGART SA851R	525	495	475
QUME DATA TRACK 8	525	495	475
MITSUBISHI M2894-63	485	475	469
OLIVETTI 802/851	369	359	349
TANDON 848-2 SLIMLINE	495	485	475
SHUGART 860 THINLINE	569	549	539
Five Inch Single Sided			
SHUGART SA400	215	209	199
TANDON TM 100-1	209	199	195
Five Inch Double Sided			
SHUGART SA450	349	329	315
TANDON TM 100-2	295	269	259
TANDON 96TPI TM100-4	369	355	350
OLIVETTI 502 ⅔ height	239	225	215
Three Inch Rigid Floppy			
HITACHI-AMDEK	call for pricing		
Five Inch Winchesters			
SEAGATE 506 6 Megabyte	759	725	695
SEAGATE 512 12 Megabyte	995	960	960
TANDON 603SE 14 Megabyte	995	960	895
WESTERN DYNAX removable	995	960	950

Upon request, all drives are supplied with power connectors and manual



$750 *Eight Inch Subsystem*

Two Siemens FDD100-8 disk drives with power supply, 4" exhaust fan complete with all necessary power cables.

Same as above but with:

Shugart 801R	MSD2801	$1195
Shugart 851R	MSD2851	1450
Olivetti 802	CAL2802	$1250
Qume DT8	MSD8DT	1450

ECLIPSE 100 $695

INDUSTRIAL S-100 MAINFRAME

Suitable for hospital and industrial applications. Constructed from 304 brushed stainless. Modular 500 watt toroid power supply provides +8volts at 30Amps and ±16volts at 4Amps. Supplied with standard 18 slot Faraday mother board. Auxilary switched AC receptacles. The Eclipse 100 can be either table or rack mounted. Provisions for internally mounting a ten megabyte Winchester disk drive.

The Eclipse 100 is the perfect mainframe to fill the void left by the now defunct TEI Corp. EPS-100 50 Lbs.

VISA | master charge

All merchandise sold by California Digital is premium grade.
Shipping: First five pounds $3.00. Each additional $.50.
Foreign orders: 10% shipping. Excess will be refunded.
California residents add 6½% sales tax. COD's discouraged.
Open accounts extended to state supported educational institutions and companies with a "strong" Dun & Bradstreet.
Warehouse: 15608 Inglewood Blvd. Visitors by appointment.

TOLL FREE ORDER LINE
(800) 421-5041
TECHNICAL & CALIFORNIA
(213) 679-9001

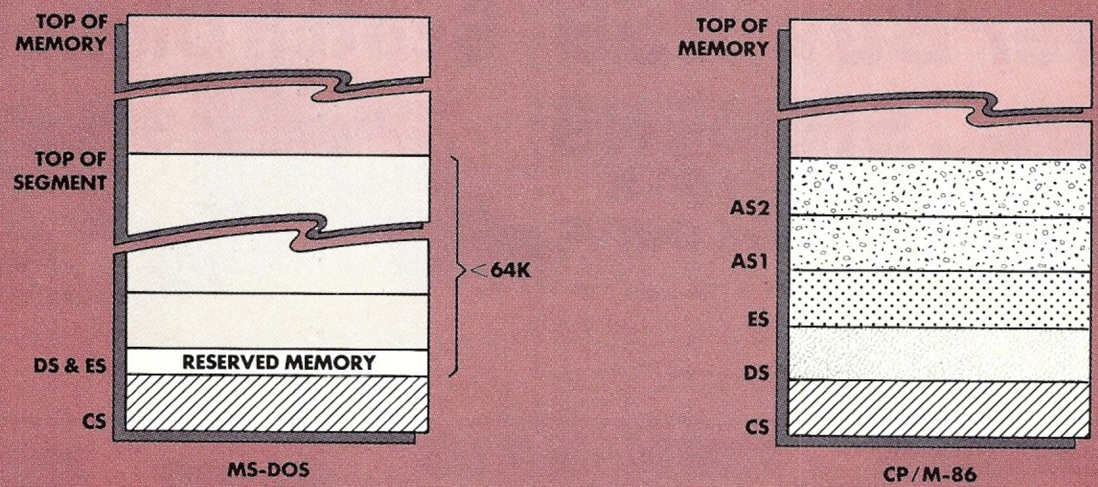


Figure 4—Memory Management Models

	CP/M-86	MS-DOS
Maximum physical sector size	—	16K
Maximum allocation blocks/volume	8K	4K
Maximum storage/volume	8M	1024M
Maximum file size	8M	1024M
Maximum directory entries	8192*	4080
Maximum memory table size/volume	1K	6K

**CP/M-86 = Allocation size/2.*

These figures show two significant features. First, MS-DOS supports much larger volume and file sizes than CP/M-86. Secondly, MS-DOS uses more memory per volume than CP/M-86. This latter point is somewhat deceptive because MS-DOS makes a substantial speed vs. space tradeoff by using the additional memory, a performance feature I will discuss shortly.

Probably the most important difference between MS-DOS and CP/M-86 file management is that they are based on entirely different file structures. MS-DOS media are not compatible with CP/M-86 media, and vice versa. CP/M-86 media are 100% compatible with CP/M-80 media, however, and there lies the crux of the matter. Since CP/M-86 is constrained to the same file structures and file management as CP/M-80, there has been no improvement or enhancement of the access or management facilities. Under CP/M-86, a file consists of 128 byte records, with 128 records/logical extent. All file access is based on these records, and end-of-file detection is also done on a record boundary. All other management of the data within a file must be done by the program.

MS-DOS, by using a new file structure, has added many new file management facilities to the system. MS-DOS files are stored as one long string of bytes, and an exact 32-bit byte count end-of-file pointer is maintained for each file. Files may be accessed as if consisting of a set of records of some program-determined size, but that size may be anywhere from 1 byte to 64K. This logical record size may be changed from access to access (it is not stored internally in the file structure or between accesses), and all accesses to the file are based upon it, including read, write, and random I/O. Note that this allows random access to a specific byte in a disk file by using a 32-bit random key and a 1-byte logical record size. In addition, there are system calls that allow the reading or writing of multiple records at one time. This multiple read call allows entire files to be read or written with one system call, and is used internally by MS-DOS for high-speed program loading.

It is important to note how the ability to use a 1-byte logical record size, along with the previously mentioned ability to use serial device names in file I/O, provides device-independent serial I/O.

In addition, MS-DOS marks all files with the time and date of each modification. This provides a much needed capability to perform incremental backup of large capacity storage media (such as Winchester disks) based upon the actual use of the files on those media.

Two performance notes concerning MS-DOS. First, MS-DOS keeps the file allocations memory

Both operating systems support an 8080 program model . . . and a model that supports separate code and data/stack segments.

ONE MAGAZINE CAN'T COVER IT ALL

THESE FOUR MAGAZINES DO!

Ziff-Davis Publishing Company is proud to introduce its ZD Computer Magazine Group. Here are the leading magazines in four widely separate areas of computing—pick the one that meets your needs and subscribe today at savings up to 25%!

■ SOFTWARE

Creative Computing is number one for software and applications information. You'll find reviews of software for all microcomputer systems, evaluations of personal computers and peripherals, book reviews and games to play. Plus helpful shortcuts, new products, and plenty of ways to have fun with your microcomputer! **SAVE 20%!**

■ HARDWARE

More people depend upon *Computers & Electronics* (formerly *Popular Electronics*) for microcomputer hardware information than *any* other magazine. You get the latest news on computing, plus advice on choosing microcomputers, peripherals and accessories. Also features on home electronics, communications, etc. **SAVE 19%!**

■ BUSINESS USES

If you're in business, sooner or later you'll need a microcomputer. *Small Business Computers* reports on (and evaluates) all the new microcomputers and operating systems for business use. You get detailed information on applications and software for finance, marketing, sales, distribution, purchasing and word processing. **SAVE 25%!**

■ ADVANCED SYSTEMS

If you're already an expert with microcomputers, you need *Microsystems,* the journal for advanced users. *Microsystems* covers new languages. Advanced hardware and software products. Tutorials on modifications and special uses of systems. Techniques for interfacing. Plus other current information on DOS development. **SAVE 20%!**

ZD COMPUTER MAGAZINE GROUP
CN 1935, Morristown, New Jersey 07960

YES! Enter my subscription(s) for the following magazines:

☐ CREATIVE COMPUTING
12 issues for $19.97.
Saving based on full subscription price of $24.97.

☐ COMPUTERS & ELECTRONICS
12 issues for $12.97.
Saving based on full subscription price of $15.97.

Mr./Mrs./Ms. ______________________
(please print full name) 8H906

Company ______________________

Address ______________________

City ____________ State ________ Zip ________

CHECK ONE: ☐ Payment enclosed. (Make check payable to ZD Computer Group.)
☐ Bill me later.

Please allow 30 to 60 days for delivery of first issue.
Offer valid in U.S. and possessions only.

☐ SMALL BUSINESS COMPUTERS
12 issues for $14.97.
Saving based on full subscription price of $19.97.

☐ MICROSYSTEMS
12 issues for $19.97.
Saving based on full subscription price of $24.97.

resident for *all* files on a given volume. This uses substantial amounts of memory (6K maximum/volume, as mentioned earlier, although the IBM-PC uses only 0.5K), but provides very fast file access. CP/M-86 keeps its indexes on disk, and may require a directory search during a file access, especially a random one. This difference of file access methodology can, in many cases, provide a performance improvement of an order of magnitude or more for MS-DOS-based programs that use many common data access techniques, including ISAM and B trees. Both systems require directory searches during file opens and closes.

Second, MS-DOS allows removable media to be changed at any time during program execution, as long as there are no currently open output files on that media. CP/M-86 requires that an explicit "reset" be issued to the drive after a media change has occurred if output is to be written to the new media; otherwise output is inhibited.

One of the major difficulties with the current MS-DOS is the lack of any form of directory partitioning. CP/M-86 provides for the partitioning of a single volume's directory into 32 different "user" areas. On volumes with large directories (greater than 128 entries), the lack of directory partitioning makes it difficult to use large numbers of files.

As a final note on the area of file management, both MS-DOS and CP/M-86 provide for "file attributes." CP/M-86 provides two attributes: "read only," and "system" (which inhibits directory listing of the file). MS-DOS also provides two: "hidden" and "system," which both inhibit directory listing of the file.

Command processing

The command processors for both operating systems have been designed to retain the flavor of CP/M-80. CP/M-86's is almost an exact copy of the CP/M-80 CCP, with no additional features. MS-DOS has provided all of the CP/M-80 capabilities, plus some new ones:

—The COPY function (the PIP program in CP/M-86) is built into the MS-DOS command processor, allowing file transfers to be made at any time without having to load a transfer program first.

—The BATCH function (the SUBMIT program in CP/M-86) is built into the MS-DOS command processor. Files with an extension (type) of .BAT are automatically submitted to the batch job stream when their file name is given as a command, in the same manner as a .COM or .EXE file would be. No explicit "submit" command is required.

—The batch file AUTOEXEC.BAT is automatically submitted to the batch job stream on an MS-DOS cold start. It may in turn execute whatever configuration or start-up programs are required, or start the execution of a turnkey application system.

—The entire command processor may be replaced simply by placing a new COMMAND.COM file on the system disk. This allows easy installation of new user interfaces to MS-DOS if required.

—The DIR command lists not only the file name but also the file size and file modification date and time.

Standard support utilities

The system support utilities that are provided as part of the operating system releases are:

CP/M-86	
ASM86*	absolute assembler
GENCMD*	executable program builder
STAT	disk/file/device utility
PIP	file transfer utility
DDT86	machine language debugger
ED	character-oriented text editor
SUBMIT	batch processing utility
LDCOPY	boot loader copy utility
misc*	system configuration and build utilities

**provided in both 8080 and 8086 versions*

MS-DOS	
M86	relocatable macro assembler
DEBUG	machine language debugger
CHKDSK	disk utility
SYS	operating system copy utility
RDCPM	file transfer from CP/M-80 to MS-DOS
EDLIN	line-oriented text editor
LINK	linkage editor
FILCOM	file comparison utility
TRANS	8080/Z80 to 8086 translator
EXE2BIN	EXE to COM converter
C86	cross reference utility
LIB	library manager

Although another whole paper could be written on these utilities, some comments are perhaps useful. The CP/M-86 assembler uses mnemonics that are close to the Intel standard, with some minor variations, and generates only absolute (nonlinkable) output. The MS-DOS assembler is 100% Intel compatible, with a full macro capability, and generates relocatable and linkable output.

The debuggers provided with each system are similar, with some minor differences. CP/M-86's DDT-86 has the ability to do in-line assembly

MS-DOS provides a number of features not available in CP/M-86 for use by the program in controlling its execution process.

while debugging, but MS-DOS's DEBUG does not. DEBUG has a memory search option, and a direct disk I/O (by specific sector), while DDT-86 does not.

Both systems provide basic text editors that are normally used as little as possible, since there are many screen-oriented editors of much greater sophistication available for both systems. Both editors are usable if necessary, however.

MS-DOS provides a linkage editor for use with M86 as well as the many other languages supported by Microsoft. The provision of a standard linkage editor will hopefully motivate language developers to produce one relocatable file format for the MS-DOS environment, in this case the Intel standard.

MS-DOS also provides an 8080/Z80 to 8086 assembly language translator that accepts M80-compatible source files and produces M86-compatible ones. Digital Research markets a similar product, but it is available only at an additional cost.

One final utility that is unique to the MS-DOS system is FILCOM, a sophisticated file comparison utility that can locate the changes (additions, deletions, and modifications) between two ASCII files and report them. It can also do binary file comparisons.

CP/M conversion effort

Probably the most asked question about these two operating systems concerns their respective "upward compatibility" from CP/M-80. From my point of view, the major issues in this area are disk compatibility, system call compatibility, memory image compatibility, and performance.

I have already discussed the fact that MS-DOS disk structures are incompatible with CP/M-80's, while CP/M-86's are identical. This means that a conversion of all data files to the new format is necessary to convert to MS-DOS. A utility program, RDCPM, is provided to do this, but this conversion can be time-consuming and difficult (especially with large data files).

The issues of system call and memory image compatibility are really the concern only of assembly language programmers. These issues are masked by the runtime environment provided by high-level languages. Since most of my company's software is in assembler, however, we found the matter one of great concern. Both systems maintain a high degree of system call compatibility. CP/M-86 does require that all CALLs and JUMPs to addresses 5H and 0H be changed to an INT instruction. MS-DOS also uses the INT instruction as the preferred way of making a system call, but it does support a CP/M-80 compatible

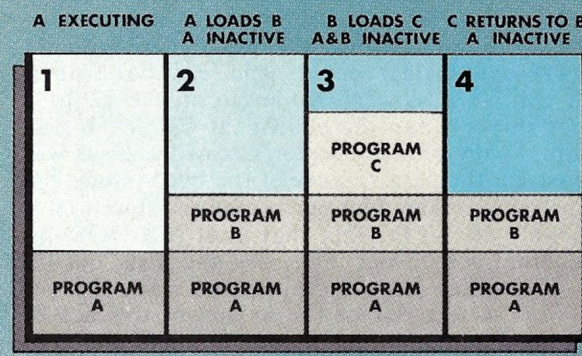


Figure 5—Program Stacking

mode of system calls at locations 5H and 0H. The use of registers during system calls is similar on both systems, and does allow for easy mechanical translation from 8080 to 8086 code. It should be realized, however, that system calls constitute only a small portion of the total code in any given program.

It should be noted that the emulation of CP/M-80 calls is not perfect under either system, and that slight inconsistencies could cause specific programs that worked perfectly under CP/M-80 to fail after translation, even though logically identical. This has been only a minor problem to our conversion efforts, since the number of differences have been small and the affected programs are easy to identify.

Another incompatibility at the system call level is the absence of a compatible direct BIOS call capability by the use of locations 1H and 2H in the memory image. CP/M-86 does provide a means by which its BIOS can be accessed directly, but the code to do so must be rewritten. MS-DOS provides no direct BIOS access.

Both MS-DOS and CP/M-86 do emulate the memory image of a CP/M-80 system remarkably well. The reserved memory locations between 0H and FFH are used in just the same manner as in CP/M-80, including the command buffer at 80H, the default FCBs at 5CH and 6CH, and the top-of-memory value at 6H-7H. Programs that rely on these locations can be converted with virtually no change.

The performance issues of conversion are interesting. Since CP/M-86 uses the same disk format and access techniques as CP/M-80, a converted CP/M-80 program should exhibit approximately the same I/O performance as was experienced under CP/M-80.

MS-DOS, however, uses an entirely different

Application conversion . . . has provided us with valuable insights into the strengths and weaknesses of both systems.

disk access and blocking/deblocking technique, and our experience has shown that directly converted programs that use CP/M-80-compatible I/O (128-byte logical records, one record at a time) will exhibit a slower I/O performance (2 to 3 times slower) than the equivalent CP/M-86 program. As the physical sector size of the disks was increased, the performance of the two systems became very similar. This performance differential is probably due to the fact that most MS-DOS implementations do not interleave the disk, but instead optimize for the multisector reads. Larger physical sectors reduce the impact of the lack of interleaving, hence the noted result. If the converted programs are modified to use larger disk buffers internally, with the CP/M-86 program doing multiple sequential 128-byte reads to fill the buffer and the MS-DOS program doing a single I/O call to fill the buffer, the MS-DOS program will run three to four times faster than the CP/M-86 one. This performance increase due to multiple record I/O is automatically available to users of high-level languages, because the runtime support for these languages under MS-DOS has been designed to use these new capabilities.

Reliability

Reliability is always a touchy issue. Who is to say when one new software system is more reliable than another? I can only relate our experience to date. Our own experience has shown a vast reliability difference between the two systems. We have been using CP/M-86 for almost a year, in a number of different environments, and through two different releases, and are still constantly finding bugs in the system, including complete features that just do not work as documented. We have been using MS-DOS for only about four months, but have yet to find a single bug in the system. This is not to imply that there are not any, only that in normal use, none have been experienced by our staff.

I realize that any assessment of this sort is very subjective, so certainly take it with as many grains of salt as is appropriate.

Conclusions

I am not sure that any objective conclusions can easily be drawn from this presentation. Every potential user of the two operating systems has his own requirements for what an operating system should supply in the way of capabilities and performance. What I have attempted to present here are the major differences and, by implication, the major similarities between CP/M-86 and MS-DOS. Neither of these operating systems has yet reached a plateau of maturity. It is obvious, from the announced plans of both Microsoft and Digital

Programmer's continued . . .

Research, that many of the areas discussed here are being addressed in upcoming releases of both operating systems.

My own experience with these two operating system may perhaps be of interest. Our development staff, which has been designing and implementing CP/M-80 based software for five years and is now making the transition to the 8086, has expressed a decided preference for the MS-DOS environment. MS-DOS is a new system, started entirely from scratch, and embodying a new conceptual base and new approaches. The transition to MS-DOS therefore requires more effort on the part of a programmer than the one from CP/M-80 to CP/M-86. Since all of my software staff are now using MS-DOS for their development environment (a choice not imposed upon them), I must conclude that MS-DOS provides a more productive environment in which to work, and a better foundation upon which to build new software products.

Neil Colvin is President of Phoenix Software Inc., N. Easton, MA. He and his associates have been creating software for microcomputers for about 8 years. He is probably best known for the software he created for TDL (Technical Design Labs).

U.S. MICRO SALES

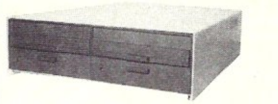
8" DISK DRIVES

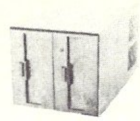
SA801R	388.00ea	Two for 379.00ea
SA851R	535.00ea	Two for 529.00ea
QUME DT-8	540.00ea	Two for 529.00ea
Tandon 848-1	395.00ea	Two for 388.00ea
Tandon 848-2	525.00ea	Two for 519.00ea

We carry a full Line of 5¼" Drives and Subsystems.

CALL TOLL FREE FOR PRICES!

DUAL DRIVE SUBSYSTEMS


HORIZONTAL


VERTICAL

w/two Shugart 801R SS/DD	$ 975.00
w/two Shugart 851R DS/DD	1225.00
w/two Qume DT-8 DS/DD	1250.00
w/two Tandon 848-1 SS/DD	995.00
w/two Tandon 848-2 DS/DD	1195.00
Cabinet A & T w/Power Supply and Accs	235.00
Cabinet Top and Bottom	69.50

California Computer Systems SPECIAL!

CCS SYSTEM 2410 . . $1995.00

- Includes CP/M® 2.2
- 2-Serial/1-Parallel Port
- DMA Disk Controller
- Hardware Vectored Interupts
- 2-Real Time Clocks
- Supports CP/M®, MP/M®, OASIS

CCS 2200 System, A & T . . 1625.00

- 2810 CPU .... *Only*—255.00
- 2422 Disk Controller .... *Only*—330.00
- 2065C 64K .... *Only*—320.00
- 2200A Mainframe .... *Only*—475.00
- CCS Apple Boards . . . **Call Toll Free For Prices**

SOFTWARE

BASIC 80	$290.00
BASIC COMPILER	320.00
CALCSTAR	150.00
C BASIC	118.00
C COMPILIER	175.00
COBOL 80	600.00
DATA STAR	265.00
DISK DOCTOR	85.00
FORTH	170.00
FORTRAN 80	400.00
MACRO 80	165.00
MACRO ASSEMBLER	90.00
MAILMERGE	115.00
PASCAL/M	320.00
SPELLSTAR	190.00
SUPERCALC	235.00
SUPERSORT I	190.00
SSS FORTRAN	215.00
WORDSTAR	275.00

EXTRA MANUALS AVAILABLE

PRINTERS

ANADEX 9001A	$ 1225.00
ANADEX 9501A	1325.00
EPSON MX-80	449.00
EPSON MX-80FT	559.00
EPSON MX-100	735.00
I.D.S. MICROPRISM 480	565.00
OKIDATA 82A	465.00
OKIDATA 83A	750.00
OKIDATA 84	1275.00
PROWRITER 8510 Par	525.00
PROWRITER 8510 Ser	625.00

TERMINALS

ADDS VIEWPOINT........$519.00
Green Screen, Detachable Keyboard

ADDS REGENT 20	569.00
ADDS REGENT 25	745.00
ZENITH Z-19	745.00

TELEVIDEO

★ THE NEW **910+** $569.00
Green Screen, Block Mode Terminal and 10 Preprogrammed Function Codes.

★ THE NEW **925** $739.00
With 22 Function Keys, Green Screen and Detachable Keyboard.
• *Time of Day* • *Self Test* • *And More!* •

★ THE NEW **950** $945.00

CALL FOR ***QUANTITY*** *PRICES!*

APPLE®, ATARI® and TRS-80® OWNERS!

Three fantastic books from *Creative Computing*...

THE CREATIVE APPLE

The best articles and features on the Apple to appear in the past four years of *Creative Computing* are gathered together in this interesting and entertaining book. Revised and updated to include the latest information, this wealth of material gives you dozens of ways to tap the tremendous potential of your Apple. Contents range from graphics, music and educational applications to word processing, software reviews, games and tips on easier programming. Whatever you use your Apple for, you'll find plenty here to help you get more out of it than you ever imagined possible.

The Creative Apple, edited by Mark Pelczarski and Joe Tate. 8½" x 11", softcover, 450 pages, illustrated. $15.95.

THE CREATIVE ATARI

Whether you're an Atari beginner or seasoned expert, you really shouldn't be without this extraordinarily useful guide to practical Atari "how to" and lore. You get tutorials on Atari graphics, a discussion of Atari memory concepts, programming tips, programs and much more that will help you make better use of your Atari. It's written so the average, non-expert user who knows a little about BASIC and simple programming can easily develop into a skilled Atari user. The contents have appeared as articles, columns and tutorials in *Creative Computing,* but have been thoroughly revised and updated to insure maximum learning ease.

The Creative Atari, edited by David Small, Sandy Small and George Blank. 8½" x 11", softcover, 250 pages, illustrated. $15.95.

THE CREATIVE TRS-80

Here's a complete guide to everything you can do with your TRS-80 and how to do it better, faster and more expertly: games, graphics, educational applications, business applications, programming aids, music and a review of handy hardware and peripherals. Whether you're interested in expense management or educational techniques, symphonies or color composition, this valuable volume will enable you to use your TRS-80 for more exciting and satisfying computing almost immediately.

The Creative TRS-80, edited by Ken Mazur. 8½" x 11, softcover, 250 pages, illustrated. $15.95.

Registered Trademarks: Apple: Apple Computer, Inc.; Atari: Atari, Inc.; TRS-80: Radio Shack division of Tandy Corp.

Use this handy coupon, or the toll-free number below, to order today!

creative computing Dept.NA3H, 39 East Hanover Avenue Morris Plains, NJ 07950

CREATIVE APPLE: Send me ________ copies at $15.95 each* plus $2 postage and handling. #18R

CREATIVE ATARI: Send me ________ copies at $15.95 each* plus $2 postage and handling. #18B

CREATIVE TRS-80: Send me ________ copies at $15.95 each* plus $2 postage and handling. #18Y

All foreign orders (except Canada and Mexico) add $3 to regular postage and handling charge. Shipped airmail only. *NJ residents add 5% sales tax.

☐ PAYMENT ENCLOSED $________

☐ CHARGE MY: ☐ American Express ☐ MasterCard ☐ Visa

Card No.________________ Exp. Date________

Signature________________

Mr./Mrs./Ms.________________ (please print full name)

Address________________ Apt.______

City________ State________ Zip________

Institutions only—Purchase orders under $50 not accepted.

For faster service, **PHONE TOLL FREE: 800-631-8112** (In NJ call 201-540-0445)

An Ada Tutorial, Part II

by Mark M. Zeiger

Types

One of the strengths of Pascal is the number of different "types" the programmer has or may create. Ada has copied much of this and added some refinements of its own.

Numbers

Ada has provided for the predefined types *integer* and *real*. The maximum and minimum values of integers and reals are *implementation dependent*. Identifiers must be declared to be one or the other, and the way a number is written will specify which type of number it is. If we have:

```
I : integer;
R : real;
```

then R := 21 is illegal. It must be written R := 21.0. Naturally I := 21.0 would be illegal. Integers may use the "E" notation, so that I := 21E2 is a valid Ada statement and assigns 2100 to I.

Numbers may also include an underscore for readability, so that 27_421 is a legal number. The underscore may not be adjacent to the decimal point; therefore 247._123 is syntactically incorrect. Ada also provides for number systems other than decimal (up to base 16). The syntax is:

(base in decimal)#(legal digits of base)#

The decimal number 12 may be written in binary as 2#1100#, in octal as 8#14#, and in hex as 16#0C#. The based number may have an exponent whose base is the same as the declared base. Thus 3#21#E21 is 7*(3**21). Note that the exponent is always assumed to be decimal no matter what base is being used; therefore 3 is being raised to the twenty-first power and not the seventh power.

Operations between reals and integers are not allowed. If such operations are needed, then the conversion functions "REAL" and "INTEGER" must be used. Consider the declarations

```
I, J : integer;
Q, R : real;
```

If Q contains 21.0, then I := Q and Q := I are illegal and must be done as

```
I := INTEGER(Q);
```

and

```
Q := REAL(I);
```

respectively. If we wanted the sum of I and Q in R, it must be done as

```
R := REAL(I) + Q;
```

The conversion from *real* to *integer* rounds rather than truncates, so INTEGER(43.7) becomes 44 instead of the expected 43.

As in Pascal, constants may be declared. In Pascal, constants must be predefined types such as "boolean," "real," "integer," and "character," since constant declarations must come before all others. In Ada, constants may be of any type. Therefore the following may be found in an Ada program:

```
ONE : constant integer := 1;
R_ONE : constant real := 1.0;
type COLOR is (RED, GREEN, BLUE, ORANGE);
MY_FAVORITE_COLOR : constant COLOR := BLUE;
              -- illegal in Pascal
```

In the case of numeric constants only, the type may be omitted, since it can be derived from the presence or absence of the decimal point.

Enumeration types

As seen in many of the previous examples, Ada implements the enumeration types found in Pascal. Ada also provides for subtypes that are the same as a declared type but with a constraint placed upon them. If the type MONTH is declared as

```
type MONTH is (JAN,FEB,MAR,APR,MAY,
               JUN,JUL,AUG,SEP,OCT,NOV,DEC);
```

then SUMMER_MONTHS may be declared as

```
subtype SUMMER_MONTHS is MONTH range JUL..SEP;
```

We could declare variables as either the type or the subtype, and assign one to the other as in:

```
X : MONTH;
Y : SUMMER_MONTHS;
```

and the following would be legal:

```
X := Y;         -- always okay
Y := X;         -- may raise an error
```

but if X does not contain one of the SUMMER_MONTHS, then an error (called a CONSTRAINT_ERROR) will be raised. Also, Y could have been declared a subtype without defining a specific subtype as follows:

```
Y : MONTH range JUL..SEP;
```

There are a number of useful attributes associated with scalar and enumeration types. The attributes "first" and "last" return the first and last element of a type. The syntax and values are:

```
MONTH'first            -- is JAN
MONTH'last             -- is DEC
SUMMER_MONTHS'first    -- is JUL
integer'first       -- is -32768 on a 16-bit
                    --    integer implementation
integer'last        -- is +32767 if 16-bit integer
```

There are also the "pred" and "succ" attributes along with the "val" and "pos" attributes:

```
MONTH'succ(JAN)        -- is FEB
MONTH'pred(APR)        -- is MAR
MONTH'pos(JAN)         -- is the integer 0
MONTH'pos(DEC)         -- is 11
MONTH'val(3)           -- is APR
```

The following will all raise the CONSTRAINT_ERROR:

```
MONTH'succ(DEC)
SUMMER_MONTHS'pred(JUL)
MONTH'val(12)          -- since JAN is 0, DEC is 11
```

We can naturally take any combinations of these attributes; therefore if we declare:

```
X : MONTH := MAR;
W : MONTH
```

then

```
W := MONTH'val(MONTH'pos(MONTH'last) - MONTH'pos(X))
```

assigns OCT to W.

Elements of enumeration types may be com-

Mark M. Zeiger, 198-01B 67th Avenue, Flushing, NY 11265

pared. "JAN <Feb" would return the boolean value "TRUE", since JAN is positioned before FEB in the type declaration. And finally, the members of an enumeration type may be overloaded. Consider the following two types:

```
type DAY is (SUN,MON,TUE,WED,THU,FRI,SAT);
```

and

```
type POSITION is (STOOD,SAT,KNEELED,STRETCHED);
```

This would be illegal in Pascal (because of SAT), but it is perfectly okay in Ada, and we say that "SAT" is overloaded. Which "SAT" is to be used is usually figured out by context, but if there will be ambiguity, then we may write DAY'(SAT) or POSITION'(SAT) depending upon which "SAT" we really want to use.

Ada also has what is called the "derived type." A derived type is similar to an existing type, but is considered different from the existing type. Using the type MONTH previously defined, we may create the derived type MONTHX by:

```
type MONTHX is new MONTH;
```

If we now declare the variables

```
M  : MONTH  := FEB;
MX : MONTHX := AUG;
```

then MX := M or M := MX is illegal, since even though the types appear the same and have the same construction, they are different types. However, a variable of one type may be converted to another. If we wish to convert the current value of MX, which is of type MONTHX, to type MONTH, we may write MONTH(MX). This is similar to the instance when we used the construct INTEGER(Q). It is now permissible to have

```
M := MONTH(MX);
```

or

```
MX := MONTHX(M);
```

A derived type takes on the same structure as its "parent" type, so we may have derived types of scalars, arrays, records, or any other Ada structures.

Booleans

Booleans are predefined enumeration types of (false, true). The Boolean operators are "not", "and", "or", and "xor" (exclusive "or"). There are a few differences between the way Booleans are used in Pascal and Ada. Since "and", "or", and "xor" have lower precedence than any of the relational operators (=, >, etc.), parentheses are not needed in a statement such as

```
if A = B and C = D then
```

as they are in Pascal. However, parentheses are needed when mixing the operators, so "Q and R or S" is illegal and must be written either as "(Q and R) or S" or as "Q and (R or S)". Since "not" has a higher precedence than the other operators, "not Q or R" means "(not Q) or R".

When Boolean expressions are evaluated in an "if" statement, all parts of the expression are evaluated. This can lead to problems. Consider the following conditional:

```
A : array (1..20) of integer;
I : integer;
if I > 0 and A(I) = 40 then
```

The above Boolean expression will raise CONSTRAINT_ERROR if I is less than zero (since the array has a range of 1..20), even though it seems that a test was included to make sure I is greater than zero. The reason is that both parts of the Boolean are evaluated, even if the Boolean's value can be determined from the first part of the statement. So the statement will attempt to check the value of something such as A(0), raising an error since A(0) does not exist. However, the creators of Ada anticipated this problem and included operators to handle it. Instead of using "and", we may use the operator "and then". If the above statement is coded as

```
if I > 0 and then A(I) = 40 then
```

then the second part of the Boolean expression will not be evaluated unless the first part is true. Ada also has the "or else" construct. If B1 and B2 are Boolean variables, then in evaluating the expression B1 or else B2, if B1 is true, B2 will not be evaluated since logically the entire expression is true not matter what the value of B2.

Ada also has membership tests that may be used to see if a value is in a certain range. The following are Boolean expressions that may be used in conditional statements:

```
I : integer := 20;
V : MONTH := OCT;                    -- as declared in the
                                     --   last section

if V in MONTH range JAN..MAY then        -- is false
if V not in MONTH range JAN..MAY then    -- is true
if I in range 1..100 then                -- is true
if I not in range 1..100 then            -- is false
```

Arrays

We have already worked with arrays in previous examples, but we have not seen any of the attributes associated with them. Arrays may be of several dimensions and, as in Pascal, the index may be an enumeration type. Therefore the following types are allowed:

```
type TWO_D is array (1..5,0..3) of real;
        type BYEAR is
            array (MONTH range JAN..JUN) of boolean;
        type YEAR is
            array (MONTH) of integer range 1..31;
                         -- YEAR is an array of (JAN..DEC)
```

In addition, the dimensions may be assigned dynamically. So if

```
K : integer := 30;
```

is declared, then

```
ONE_D : array (1..K) of MONTH range JAN..AUG;
```

creates the array

```
ONE_D(1)..ONE_D(30)
where each ONE_D(I) may hold
JAN..AUG.
```

The attributes "first" and "last" may be used with arrays. These attributes evaluate to the indices of the array. Therefore YEAR'first would equal JAN and BYEAR'last would equal JUN.

The "length" attribute gives the number of in-

One of the strengths of Pascal is the number of different datatypes. . . . Ada has added some refinements of its own.

dices in the first dimension of an array, so that YEAR'length is equal to 12 and TWO_D'length is equal to 5.

The "range" attribute returns the range of an index; therefore YEAR'range is equivalent to JAN..DEC. This is extremely useful in iteration loops because of instead of having to write

```
for I in JAN..DEC loop
    .
    .
end loop;
```

we may write

```
for I in YEAR'range loop
    .
    .
end loop;
```

Each of the above attributes may apply to any of the dimensions in a multidimensioned array. If we want the length of the second dimension of TWO_D, we may write

```
TWO_D'length(2)
```

which would be equivalent to four (0..3). If no dimension is specified, the default is the first dimension. Thus

```
for I in 1..5 loop
     for J in 0..3 loop
          X(I,J) := ...     -- where X is of type TWO_D
     end loop;
end loop;
```

may be replaced with

```
for I in TWO_D'range(1) loop
     for J in TWO_D'range(2) loop
          X(I,J) := ...
     end loop;
end loop;
```

The (1) in the above example is optional, but since this is a multidimensioned array, it is best to include it for clarity.

We have already seen how to assign initial values to single-dimensioned arrays. To initialize a multidimensioned array, we may write

```
X : TWO_D := ( ( 1.0,  2.0,  3.0,  4.0,  5.0)
               ( 6.0,  7.0,  8.0,  9.0, 10.0)
               (11.0, 12.0, 13.0, 14.0, 15.0) );
```

If an array is going to be a constant array, then initialization may even be easier. If the array MOY is to contain the number of days for each month, then

```
MOY : constant YEAR := (FEB => 28,
                        SEP ¶ APR ¶ JUN ¶ NOV => 30,
                        others => 31);
```

In the above example, MOY(JAN), MOY(MAR), MOY(MAY), etc. will be assigned 31.

Arrays of the same type may be copied by simple assignment statements. If we declare

```
X, Y : YEAR;
```

then X := Y is legal. Note, however, that if we have the declarations

```
A : array (1..10) of integer;
B : array (1..10) of integer;
```

then A and B are not the same type, and A := B will raise CONSTRAINT_ERROR. Of course if we write

```
A, B : array (1..10) of integer;
```

then A and B are of the same type and A := B is legal.

Other types

Like Pascal, Ada also has records, variant records, and dynamically allocated storage (called "access types" in Ada). Since the use of these types in Ada is not that much different from Pascal usage, only syntax will be described.

A record type may be declared as

```
type COMPLEX is
     record
          REAL_PART : real;
          IMAG_PART : real;
     end record;
```

A record of this type may be declared and given initial values if desired by

```
X : COMPLEX := (0.0, 0.0);
```

Just as in Pascal, each component of the record may be named using the "dot" notation. If we write X.REAL_PART := 5.7, then X will now be equivalent to 5.7 + 0i. Ada also allows us to assign values to components of a record without using dot notation. X := (4.0, 2.0) yields 4.0 + 2.0i and subsequently, X := (REAL_PART => 7.2) will make X equal to 7.2 + 2.0i. In the second example we again see the use of "named notation."

The access type of Ada is similar to that of Pascal. Let's say we wish to create a binary tree data structure. We first must have a forward reference to the data structure, then declare its access type, and finally define the full structure. Note that the forward reference is not necessary in Pascal. In Pascal our declarations would be as follows:

```
type
     LINK = ©NODE;     ξno forward reference
                           to NODE needed †

     NODE = record
          DATA : integer;
          L, R : LINK;
     end;
```

while in Ada we would write

```
type NODE;       -- the forward reference
type LINK is access NODE;
type NODE is
     record
          DATA : integer;
          L, R : LINK;
end record;
```

Any variables of an access type are automatically initialized to null. If we want to create a NODE, we use the command

```
PTR : LINK;          -- first get a variable of type LINK
PTR := new NODE;     -- a NODE is created with PTR
                     --    containing its address
```

The components of NODE are referenced by dot notation (i.e., PTR.DATA, PTR.L, OR PTR.R). When NODE is created, PTR.L and PTR.R are automatically assigned the values of null, since they are access types. Other types in the record are not initialized unless there are specific instructions to do so. The initialization could take place after NODE has been created (i.e., PTR.DATA := 4;) or upon creation (i.e., PTR := new NODE (4, null, null). If initialized upon creation, then values must be provided for each component of the record.

Ada provides for subtypes . . . with a constraint placed upon them.

Why use other computer media when you could be using MEMOREX high quality error free media?

Free Memorex Mini-Disc Offer – Get free discs!
You'll save money when you buy Memorex, because every carton of 10 Memorex 5¼ inch mini-discs sold by Communications Electronics has a coupon good for a free Memorex mini-disc. For every case of 100 Memorex mini-discs you buy from CE, you'll get 10 free Memorex mini-discs, directly from Memorex. The more you order, the more you save. Offer expires December 31, 1982. All Memorex flexible discs sold by CE are of the highest quality, certified 100% error free and backed by a full one year factory warranty.

Flexible Disc Quantity Discounts Available
Memorex Flexible Discs are packed 10 discs to a carton and 10 cartons to a case. Please order only in increments of 100 units for quantity 100 pricing. We are also willing to accommodate your smaller orders. Quantities less than 100 units are available in increments of 10 units at a 10% surcharge. **Quantity discounts** are also available. Order 500 or more discs at the same time and deduct 1%; 1,000 or more saves you 2%; 2,000 or more saves you 3%; 5,000 or more saves you 4%; 10,000 or more saves you 5%; 25,000 or more saves you 6%; 50,000 or more saves you 7% and 100,000 or more discs earns you an 8% discount off our super low quantity 100 price. Almost all Memorex Flexible Discs are immediately available from CE. Our warehouse facilities are equipped to help us get you the quality product you need, when you need it. If you need further assistance to find the flexible disc that's right for you, call the Memorex flexible disc compatibility hotline. Dial toll-free 800-538-8080 and ask for the *flexible disc hotline* extension 0997. In California dial 800-672-3525 extension 0997. Outside the U.S.A. dial 408-987-0997 between 9 AM to 4 PM Pacific Time.

SAVE ON MEMOREX FLEXIBLE DISCS Product Description	Part #	CE quant. 100 price per disc ($)
8" SSSD IBM Compatible (128 B/S, 26 Sectors)	**3062**	2.09
8" SSSD Shugart Compatible, 32 Hard Sector	**3015**	2.09
8" SSSD CPT 8000 Compatible, Soft Sector	**3045**	2.99
8" SSDD IBM Compatible (128 B/S, 26 Sectors)	**3090**	2.74
8" DSDD Soft Sector (Unformatted)	**3102**	3.34
8" DSDD Soft Sector (128 B/S, 26 Sectors)	**3115**	3.34
8" DSDD Soft Sector (256 B/S, 26 Sectors)	**3103**	3.34
8" DSDD Soft Sector (512 B/S, 15 Sectors)	**3114**	3.34
8" DSDD Soft Sector (1024 B/S, 8 Sectors)	**3104**	3.34
5¼" SSDD Soft Sector w/Hub Ring	**3481**	2.34
5¼" SSDD 10 Hard Sector w/Hub Ring	**3483**	2.34
5¼" SSDD 16 Hard Sector w/Hub Ring	**3485**	2.34
5¼" DSDD Soft Sector w/Hub Ring	**3491**	3.09
5¼" DSDD 10 Hard Sector w/Hub Ring	**3493**	3.09
5¼" DSDD 16 Hard Sector w/Hub Ring	**3495**	3.09
5¼" SSQD Soft Sector w/Hub Ring (96 TPI)	**3504**	2.99
5¼" DSQD Soft Sector w/Hub Ring (96 TPI)	**3501**	3.99

SSSD = Single Sided Single Density; SSDD = Single Sided Double Density; DSDD = Double Sided Double Density; SSQD = Single Sided Quad Density; DSQD = Double Sided Quad Density; TPI = Tracks per inch.

Special offer on Memorex computer tape.
If you mail your order to us and enclose prepayment, deduct $1.00 per reel from our quantity 100 prices. This means Memorex 25JW can be as low as $12.99 in 100 quantities. Memorex Computer Tapes are packed 10 tapes to a carton. Please order only in increments of 100 units for quantity 100 pricing. Quantities less than 100 units are available in increments of 10 units at a 10% surcharge. **Quantity discounts** are also available. Order 500 or more tapes at the same time and deduct 1%; 1,000 or more saves you 2%; 2,000 or more saves you 3%; 3,000 or more saves you 4%; 4,000 or more saves you 5% and 5,000 or more tapes earns you a 6% discount off our super low quantity 100 price. If you need further assistance or information to find the tape that's right for you, call the Memorex Computer Tape Technical Support Group at (408) 987-2937.

SAVE ON MEMOREX COMPUTER TAPE Product Description	Part #	CE quant. 100 price per reel ($)
Memorex IV 2400 feet Wrightline Seal	**25JW**	13.99
Memorex IV 2400 feet Easy Load II Cartridge	**25JR**	14.99
Memorex IV 1200 feet Wrightline Seal	**25FW**	10.65
Memorex Quantum 2400 feet Wrightline Seal	**27JW**	16.20
Memorex Quantum 2400 feet Easy Load II Cart.	**27JR**	16.99
Memorex Quantum 1200 feet Wrightline Seal	**27FW**	12.50
Memorex Cubic HD 2400 feet Wrightline Seal	**39JW**	18.99
Memorex Cubic HD 2400 feet Easy Load II	**39JR**	19.99
Memorex Cubic HD 1200 feet Wrightline Seal	**39FW**	13.99

New Memorex Lifetime Rigid Disc Pack Product Warranty
All Memorex disc packs sold by CE have a lifetime product warranty. This is your assurance that Memorex disc packs will give you a lifetime of performance and service. Only Memorex can offer you the superior reliability of their exclusive M Formula. In addition, Memorex will assist the original user in isolating and correcting any technical issues that relate to the Memorex product as well as, when appropriate, replace up to one set of read/write heads. If you need further information to find the rigid disc that's right for you, call the Memorex rigid disc compatibility hotline. Dial toll-free 800-538-8080 and ask for the *rigid disc hotline* extension 1642. In California dial 800-672-3525 extension 1642. Outside the U.S.A. dial 408-987-1642.

SAVE ON MEMOREX RIGID DISC PACKS Product Description	Part #	CE quant. one price per pack ($)
Mark III 5 MB. Cartridge Front Load (8 to 32 Sect.)	**95-522XX-03**	65.00
Top Load (1-to 24 sectors)	**94-522XX-03**	70.00
CMD-16 "Phoenix Type" CDC Cartridge	**98-26600-31**	160.00
NCR Cartridge	**98-26600-32**	160.00
Mark VIII 80 MB. Error Free	**72-16600-03**	330.00
Flag Free	**72-26600-03**	320.00
Mark XI 200 MB. Error Free	**03-35041**	720.00
Flag Free	**03-35031-02**	560.00
DEC Flag Free	**03-35031-03**	560.00
Mark XII 200 MB. NCR/CDC Flag Free	**03-39001-01**	515.00
Honeywell Flag Free	**03-39000-01**	515.00
Mark XIII 300 MB. Error Free	**03-47021**	795.00
Flag Free	**03-47009**	670.00
Mark XIV 80 MB. Unformated Error Free	**74-16600-03**	365.00
Flag Free	**74-26600-03**	300.00
Honeywell Format Flag Free	**74-26600-08**	315.00
CDC Format Flag Free	**74-26600-09**	315.00
Mark XV 300 MB. Error Free	**03-49011**	825.00
Flag Free	**03-49001-01**	725.00

Smith-Corona TP-1 Letter Quality Printer Special Offer
Buy any Memorex product on this page, and get a Smith-Corona TP-1 letter quality printer for only $585.00 plus $20.00 shipping. Specify serial or parallel version.

Buy with Confidence
To get the fastest delivery from CE of your Memorex computer products, send or phone your order directly to our Computer Products Division. Be sure to calculate your price using the CE prices in this ad. Michigan residents please add 4% sales tax. Written purchase orders are accepted from approved government agencies and most well rated firms at a 30% surcharge for net 30 billing. All sales are subject to availability, acceptance and verification. All sales are final. Prices, terms and specifications are subject to change without notice. Out of stock items will be placed on backorder automatically unless CE is instructed differently. Minimum *prepaid* order $50.00. Minimum *purchase order* $200.00. International orders are invited with a $20.00 surcharge for special handling in addition to shipping charges. All shipments are F.O.B. Ann Arbor, Michigan. No COD's please. Non-certified and foreign checks require bank clearance.

For **shipping charges** add $8.00 per case or partial-case of 100 8-inch flexible discs or $6.00 per case or partial case of 100 5¼-inch mini-discs. For tape shipping, add $1.00 per reel. For Disc packs add $10.00 per cartridge (Mark III or CMD-16) or $15.00 per disc pack for U.P.S. *ground* shipping and handling in the continental U.S.A.

Mail orders to: Communications Electronics, Box 1002, Ann Arbor, Michigan 48106 U.S.A. If you have a Master Card or Visa card, you may call and place a credit card order. Order toll-free. Dial 800-521-4414. If you are outside the U.S. or in Michigan, dial 313-994-4444. Order your Memorex computer products from CE today.

 Ad #090282

MasterCard VISA MEMBER DIRECT MAIL/MARKETING ASSOCIATION CODE OF ETHICS

Order Toll-Free! (800) 521-4414
In Michigan (313) 994-4444

MEMOREX
COMPUTER TAPE
DISC CARTRIDGES · RIGID DISCS
FLEXIBLE DISCS

COMMUNICATIONS ELECTRONICS™

Computer Products Division

854 Phoenix □ Box 1002 □ Ann Arbor, Michigan 48106 U.S.A.
Call TOLL-FREE (800) 521-4414 or outside U.S.A. (313) 994-4444

CIRCLE 51 ON READER SERVICE CARD

Exception handling

We will end our comparison with Pascal at this point, because Pascal has practically no exception handling. In fact, it is not even defined in the language, although most implementations do have some rudimentary form of error handling. Ada, on the other hand, has a very nice form of error handling—almost as nice as that of PL/I.

There are five built-in exceptions in Ada. We will discuss three of these now, and then see how to define our own exceptions. The three predefined exceptions are:

```
CONSTRAINT_ERROR  -- raised on an "out-of-range" error
NUMERIC_ERROR     -- an arithmetic error
STORAGE_ERROR     -- stack overflow, access
                  --   variable error, etc.
```

The other two errors are related to tasking, a subject beyond the scope of this article.

Exceptions are defined in blocks. These may be procedure, function, or begin-end blocks. As an example, consider

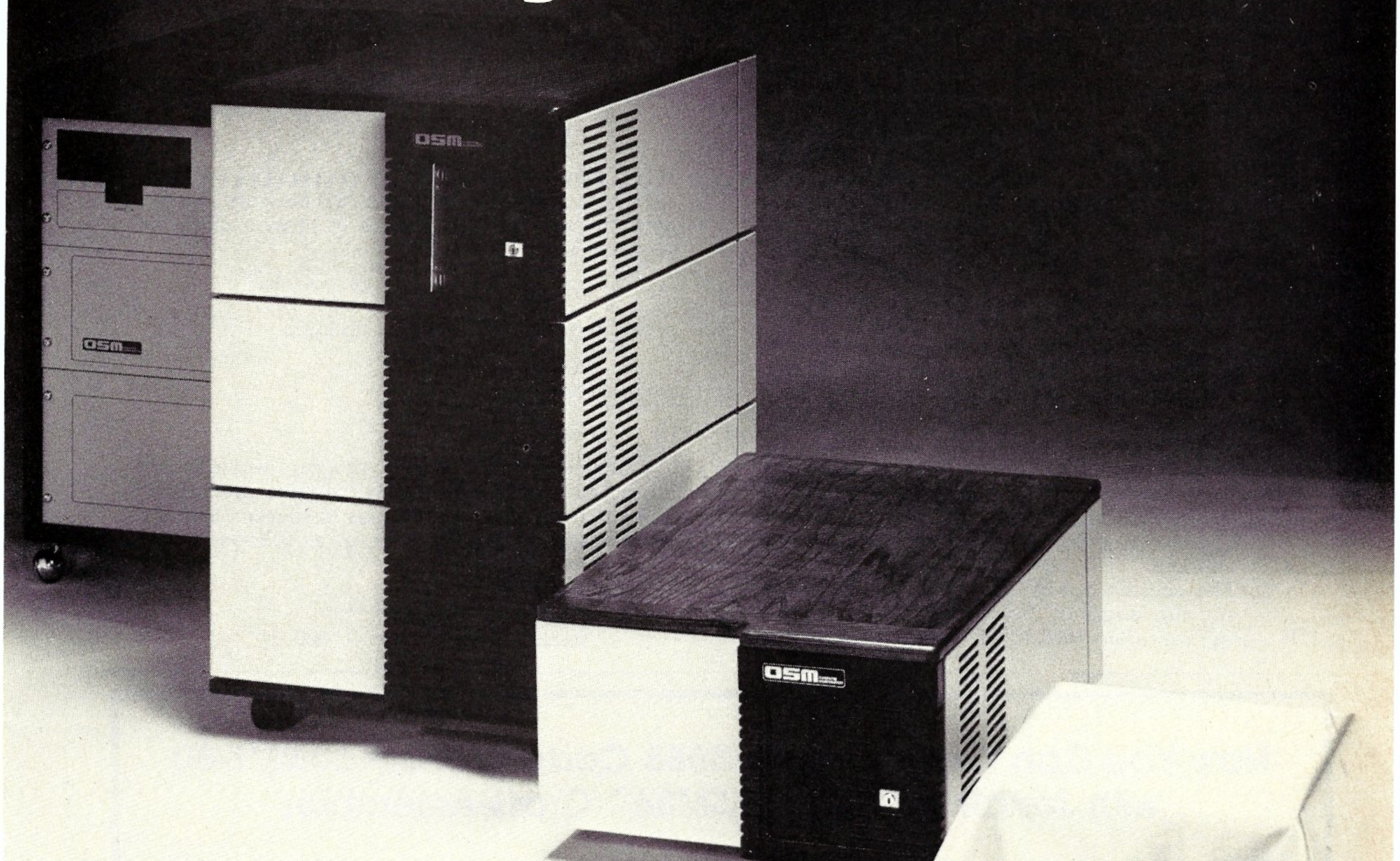
```
declare
     K : integer := 0;
begin
     loop
          K := K*2;
     end loop;
exception
     when CONSTRAINT_ERROR =>
          K := 0;
end;                    -- end of block
```

If K is going to be continually doubled, it is quickly going to be greater than integer'last (usually 32767). Without the exception statement, a runtime error would occur that would probably abort the program. With the exception, K is set to zero, and control passes to the next sequential statement after the exception clause (unless, of course, the exception clause contained a goto statement). The error handler may do anything except pass control back to the statement group that caused the error.

A programmer may define his own errors. For instance, if there were no built-in "divide-by-zero" error, we could define our own as follows:

```
declare
     type LIST is (A, B, C);
     X : LIST;
     I : real;
     J : integer;
     DIVIDE_BY_ZERO_ERROR : exception;
begin
     X := A;
     loop
          X := succ(X);     -- this will raise an error
     end loop;
          if J = 0 then
               raise DIVIDE_BY_ZERO_ERROR;
          end if;
     loop
          J := -1;
          I := I / real(J); -- this will raise an error
          J := J + 1;
     end loop;
exception
     when CONSTRAINT_ERROR =>
          X := A;
     when DIVIDE_BY_ZERO_ERROR =>
          I := 0;
     when others =>
          put("Some other error");
end;
```

OSM Offers Dealers More Ways to Grow



Having one great product to sell may keep a dealer happy for awhile. But when a dealer can offer his customers a complete family of compatible products, each one a high quality, low cost performer, then you've really got growth potential.

In 1980 OSM Computer introduced the original ZEμS1 multi-user, multi-processor microcomputer system. Aimed at the small business market, the ZEμS1 was designed for comfortable growth. Based on the Z80A microprocessor, the ZEμS1 provides each user with 64K RAM and two serial and one parallel I/O ports.

ZEμS2, our second generation product, has made significant market penetration. Easily expandable to 32 users and providing the same true multi-user multiprocessor environment, ZEμS2 is equipped with a battery backup system, up to 600MB of storage, a 20MB tape backup, and an optional 8″ floppy subsystem.

ZEμS3, introduced at NCC '82, is based on the same architecture as ZEμS2 in an 8 user configuration. Applying the latest technologies, the ZEμS3 utilizes a 5¼″ Winchester hard disk and a 5¼″ floppy drive. The cabinet measures a mere 10 x 19 x 27.5 inches, yet it contains all the hardware necessary to handle up to three shared printers as well as data communications.

OSM's MUSE operating system is CP/M* compatible and provides a true multi-user environment for ZEμS' multi-processor architecture. MUSE allows the use of more than 2000 available languages and applications programs written for CP/M* systems.

Now we are growing again. Come meet ZEμS4, the newest addition to our ZEμS family of microcomputer systems. We will be unveiling ZEμS4 at COMDEX Las Vegas booth number 2262.

Come grow with us. OSM is seeking qualified dealers/distributors. For more information regarding our company and our computer systems, please call our toll free number: 800-538-5120 and ask to speak to one of our Regional Sales Managers.

For more information, write to us:
OSM Computer Corporation, 2364 Walsh Avenue, Santa Clara, CA 95051.

OSM
Computer Corporation

2364 Walsh Avenue
Santa Clara, CA 95051
(408) 496-6910 TWX 910 338 2099

CIRCLE 100 ON READER SERVICE CARD

*CP/M is a Registered Trademark of Digital Research, Inc.

The "others" error would pertain to any of the predefined errors (except CONSTRAINT_ERROR) and any identifiers declared to be an exception that are not specifically listed in the exception handling part.

Packages

The last item I will briefly discuss is the concept of packages. In Ada it is possible to have separate programs that may be compiled separately and linked together. Since I/O is not defined in Ada, most I/O in Ada is implemented as packages unique to the particular computer being used. Since each package must know what the other contains, each package must be supplied with a package declaration that must be available to the programmer in source, and a package body that may be hidden (i.e., it may be in compiled form). To see how package declarations are made, let's assume some genius of a programmer has made up a procedure to graph line segments and circles on a plotter. He is selling his procedures in a compiled package called "PLOT". The information that must be supplied to the procedures is:

1. The two coordinates of the endpoints of the segment
2. The coordinates of the center of the circle and the length of the radius.

If a program called "GRAPH" wishes to use these procedures, then the program must be as follows:

```
package body GRAPH is
     -- declarations for this program

package PLOT is       --package specifications
     procedure CIRCLE(X, Y * integer; RADIUS : real);
     procedure LINE(X1, Y1, X2, Y2 : real);
end PLOT;

begin                      -- main program
     .
     .
     .
end GRAPH;
```

If we want to use any of the procedures, we may call them in one of two ways. We may use the dot notation and call the routine to draw a circle or segment as

```
                                -- "PLOT" is the package
PLOT.CIRCLE(2, 3, 4.0);         -- containing "CIRCLE"
PLOT.LINE(4.5, 7.0, 2.0, 9.3);  -- and "LINE"
```

or we may set up a block as follows:

```
declare
     use PLOT;
begin
     CIRCLE(2, 3, 4.0);
     LINE(4.5, 7.0, 2.0, 9.3);
end;
```

Our genius who wrote the program would be required to supply us with the compile procedures and the package specifications in source code. He would have written his package as follows:

```
package body PLOT is

.                  -- global declarations
.
procedure CIRCLE(A, B : integer; C : real) is
```



```
                  .
                  .    -- code to plot circle
                  .
        end CIRCLE;

        procedure LINE(A, B, C, D : real) is
                  .
                  .    -- code to plot line
                  .
        end LINE;

    begin               -- main part of package body
        null;           --do nothing
    end PLOT;
```

Conclusion

I hope I have given you some insight into the Ada language. It's a hugelanguage with many features that I have not discussed, since I did not want to write a book. In particular, I have not even mentioned "generics" and "tasking," which are major features of Ada.

I doubt we will ever see a full implementation of Ada on an 8-bit processor. There are, however, a few implementations of subsets of Ada running under CP/M, but they lack many of the features mentioned in this article. I think we will have to look to the 16-bit microprocessors for implementations approaching full Ada.

References

If you would like to learn more about Ada, the following books are available:

1. *Military Standard—Ada Programming Language, MIL-STD 1815*, U.S. Department of Defense. (This is the official language reference manual on Ada.)
2. Barnes, J.G.P. *Programming in Ada*, Addison-Wesley, New York, 1982. (One of the best books I ever read on a computer language.)
3. Pyle, I.C. *The Ada Programming Language*, Prentice-Hall, New Jersey, 1981. (Not one I read, but recommended by others.)
4. Wegner, Peter. *Programming with Ada*, Prentice-Hall, New Jersey, 1980. (Written before the standard was fully defined. I did not find it clear or helpful.)

Mark M. Zeiger teaches mathematics and computer science courses in a Long Island high school. He has been a computer hobbyist for the last several years, and has done a great deal of work in Assembly language in the CP/M operating system. Several of his programs may be found in the CP/M user's group library. He is also active in the Long Island Computer Association, having served as its president for a year.

Corrections

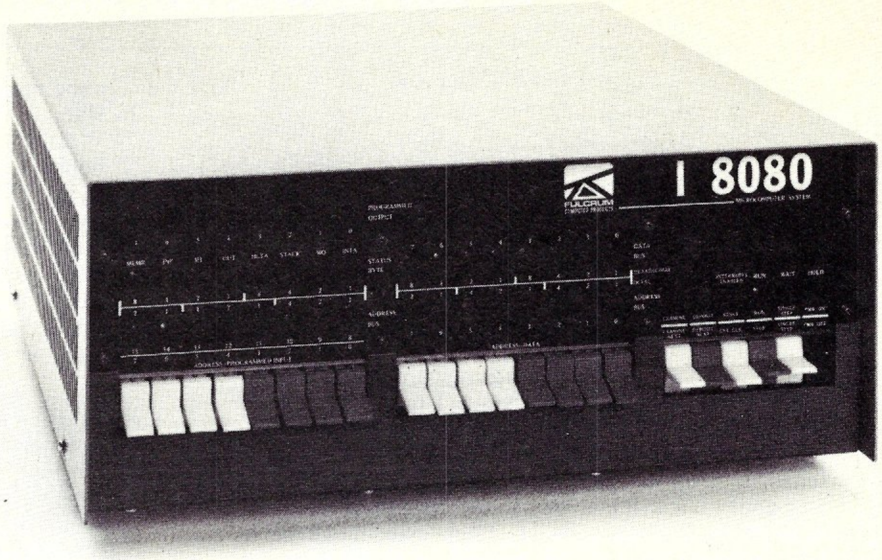
An error occurred in Part I (Sept/Oct issue) of Mark Zeiger's *Ada Tutorial*. On page 58, right column, line 16, the sentence "Once the value of X becomes less than 10 . . ." should read "Once the value of X becomes greater than 9 . . . ."

I-8080

Mainframe

The standard microcomputer includes:

- Front panel and control board (CP-A)
- Chassis with 20-slot card cage
- Sturdy, attractive dust cover (DC)
- 28-ampere power supply (PS-28)
- 10 MHz 20 slot terminated Motherboard (WB20)
- Documentation



Motherboard

Card-to-card spacing on the Mother Board is ¾-inch, except for the first position reserved for the front panel board or any other board in dedicated applications. Eight, ten, twelve and twenty slot terminated Mother Boards are available for the system, good to 10 MHz.

Heavy power traces handle the large currents that exist in a heavily loaded backplane. High-quality connectors have gold-plated contacts for reliability and long life.

Front Panel

The CP-A Board forms the operator's panel. It includes switches, indicators and logic needed for manual operation. The panel is completely self-contained and plugs directly into the first Mother Board slot. Or it may be connected through an extender board to any available slot in the Mother Board. When the first slot is not used for the front panel, that slot may be used by another board, such as the Parallel I/O Board with its LED indicators visible.

Front panel facilities include:

- 16 address/data switches
- 16 LED address indicators
- 8 LED data bus indicators
- 8 LED programmed output bit indicators
- 6 control function switches
- 8 LED status indicators (including control indicators for INTERRUPT ENABLED, RUN, WAIT and HOLD)

The front panel includes logic that drives the programmed output indicators, and reads the input byte from the high-order address switches. DATA BUS indicators show data either read or written by the processor.

Indicators are wide-angle LED's behind a contrast-enhancing acrylic panel assembly. Photographically produced panel markings are crisp and explicit and can never wear off. Bit positions are numbered and labeled for both hexadecimal and octal notation. Special labels may be easily inserted to identify special functions for the programmed output LED's.

Switches are high-quality units, with paddle handles color-coded for easy, error free operation.

Power Supply

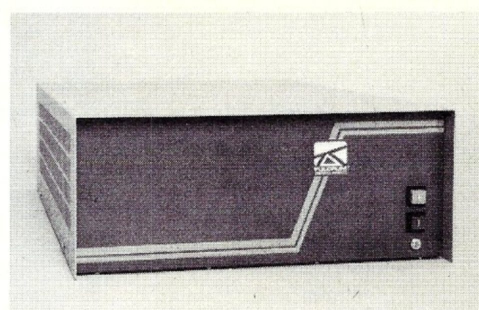
The Power Supply (PS-28) is designed for use with pc boards having on-board regulators. Outputs are +10V and ±18V at no load, and approximately +7V and ±15.8V at full load.

A Power Supply pc board contains rectifiers and 120V ac switching and fusing functions. The board provides terminals for switched ac power, both fused and unfused, for a ventilating fan and auxiliary power outlets on the back panel. When the computer is supplied without the front panel, and ac power switch is mounted on the Power Supply Board.

A custom-built transformer and large, conservatively rated filter capacitors are mounted on the chassis. All connections are made by molex connector.

8015 Mainframe

Same as 8080 mainframe except without CPA. The 8015 front has one switch for reset, one for interrupt and a key locking switch for power.



CPA	Improved Imsai style front panel works with Z80, etc.	**$225.00**
8080 Mainframe		**$630.00**
8015 Mainframe		**$475.00**

FULCRUM™
COMPUTER PRODUCTS

Distributed by:
WW COMPONENT SUPPLY INC. 1771 JUNCTION AVENUE • SAN JOSE, CA 95112 • (408) 295-7171

SERIAL I/O INTERFACE

The SIO 2-2 Serial I/O Interface board contains two identical ports, each permitting the computer to communicate with most peripheral devices through an RS232 or current loop interface. The two ports are independent. Each may operate in full-duplex or half-duplex with all control signals.

You can run synchronous or asynchronous lines, full- or half-duplex, at any baud rate up to 9600 baud (asynchronous) or 56,000 baud (synchronous). Baud rates up to 9600 (asynchronous) or 38,400 (synchronous) are selected by jumpers on the board. Asynchronous baud rates are 75, 110, 150, 300, 600, 1200, 2400, 4800 and 9600. Other rates are made possible using the SIOC board which mounts directly on the SIO board.

Control lines for each input include DSR, DTR, RTS, CTS and Carrier Detect. RS232 receivers and drivers are also provided for clocks in synchronous operations. Jumpers permit using the board as either the receiving (terminal) end of a communication line or the originating (computer) end.

Each interface is structured around an Intel 8251 USART chip. This chip allows extensive program control of I/O functions including contril line and sync character selection, and error-condition sensing and recovery. The board generates interrupts for received characters, transmitter buffer empty, transmitter empty or sync character.

The board may be jumper-adapted to respond either to I/O instructions from the I-8080 system or to memory reference instructions for memory-mapped I/O.

SIO-2-2 2 serial I/O ports, good to 9600 baud **$175.00**

IEEE-488 & 3P

The 488 + 3 is designed to be plugged into one slot of a standard IEEE-696 (S-100) cardcage. It consists of two major functional blocs:

- IEEE-488 Interface
- Three parallel ports

The IEEE-488 portion of the board is desinged to perform the interface function between an IEEE 488-1975/78 General Purpose interface Bus (GPIB) and the CPU. It communicates with the CPU via an input/output-mapped 8-bit data bus and provides a 16-bit bus to interface with the GPIB via buffer devices. IEEE 488-1975/78 standard protocol is handled automatically in Talker, Listener, and Bus or System Controller operational modes. Its specific IEEE-488 features are:

- Handles all IEEE 488-1975/78 functions
- Talker and listener functions (T, TE, L, LE)
- Automatic source and acceptor handshakes (SH, AH)
- Controller with pass control capabilities (C)
- System controller capabilities
- Device clear and trigger functions (DC, DT)
- Service request functions (SR)
- Parallel and serial poll facilities (PP, SP)
- Remote/local with local lockout (RL)
- Single or dual addressing modes
- Secondary addressing capabilities

The three parallel input/output ports portion of the board is designed to perform as a general purpose programmable I/O device. It has 24 I/O pins which may be individually programmed in two groups of twelve and used in three major modes of operation.

The 488 + 3's flexibility is enhanced by jumpers which allow the user to select input/output addresses, interrupt priorities, etc.

The software I/O driver routines supplied with the 488 + 3 facilitate its use. These programs are callable subroutines for performing message handling. The manner in which they have been written allows them to be easily incorporated into a software program.

IEEE 488 + 3 .... **$375.00**

64K STATIC RAM BOARD

Does your current 64K board have these features?

FEATURES

$425

- Conforms to IEEE 696 standard.
- 8 or 16 bit data transfers.
- 24 bit addressing.
- Bank select in 32K-32K or 48-16K.
- Banks selectable/deselectable on DMA.
- Responds to phantom pin 67 or 16.
- 2Kx8 static rams with 2716 pin out.
- Power consumption is typically 600 ma.
- Banks on or off on power up.
- Banks addressable to any of 256 possible ports.
- 8MHz with 150ns parts standard faster speeds available on request.
- Available partially loaded as a 32K board.
- Multiple bank residence.
- Phantoms read only, or read and write.
- Generates onboard M-WRITE, or uses bus M-WRITE.
- Two seperate banks, independently addressable on 16K boundaries. Banks may be overlapped.

Omniram 64 United States & Canada Prices Only

	64K	32K
150ns. Hitachi Rams	$425	$300
100ns. Toshiba Rams	$550	$395
Board Without Ram	$200	

INTELLIGENT VIDEO I/O FOR S-100 BUS

The VIO-X Video I/O Interface for the S-100 bus provides features equal to most intelligent terminals both efficiently and economically. It allows the use of standard keyboards and CRT monitors in conjunction with existing hardware and software. It will operate with no additional overhead in S-100 systems regardless of processor or system speed.

- HIGH SPEED OPERATION
- PORT MAPPED IEEE S-100 INTERFACE
- FORWARD/REVERSE SCROLL or
- PROTECTED SCREEN FIELDS
- CONVERSATIONAL or BLOCK MODE (opt)
- INTERRUPT OPERATION
- CUSTOM CHARACTER SET
- CONTROL CHARACTERS
- ESCAPE CHARACTER COMMANDS
- INTELLIGENT TERMINAL EMULATION
- TWO PAGE SCREEN MEMORY

VIO-X1 - 80 X 25 5 X 7 A & T — **$295.00**
Conversational Mode

VIO-X2 - 80 X 25 7 X 7 A & T — **$345.00**
Conversational & Block Modes

FULCRUM™
COMPUTER PRODUCTS

Distributed by:
WW COMPONENT SUPPLY INC. 1771 JUNCTION AVENUE • SAN JOSE, CA 95112 • (408) 295-7171
For European enquiries and sales contact: **Fulcrum** (Europe) England. Phone 0621-828-763

CIRCLE 10 ON READER SERVICE CARD

SOFTWARE REVIEW

Janus—A New Ada Compiler for Z80 Systems

by Harvey Fishman

Ada is a new computer language sponsored by the United States Department of Defense (DoD) for use with embedded computers in weapons systems. An embedded computer is a computer that is not visible as such to the user of the equipment. Examples would be computers that are part of fire control or display systems in weapons applications, or the microprocessors used in microwave ovens or cameras in civil applications. The name Ada is a registered trademark of the DoD, and once the language is fully implemented, will not be allowed to be applied to subsets or supersets of the language. Those implementations that are validated by DoD include all of the operations and constructs in the definition of Ada, and only those operations and constructs will be allowed to be called Ada. Ada is a very large and rich language (indeed, it has been compared with a full implementation of PL/I in this respect) and it is difficult to conceive of a compiler for the full language being implemented on an 8-bit microcomputer. This is consistent with its intent, since embedded computers do not need resident compilers, and their programs would generally be compiled using a cross-compiler on a large mainframe or super-mini. However, Ada is a language with many advantages, such as extremely powerful primitive operations, a very rich and extensible set of datatypes, and the ability to foster clear, well-structured and maintainable code. So to implement this on an 8- or 16-bit micro requires that only a subset be used, but such a subset cannot legally be called Ada.

Janus: An Ada-like compiler

Janus by RR Software is an implementation of a healthy and useful subset of the Ada language. It is offered in versions for the 8080/Z80 and for the 8086 processors. Only the 8080/Z80 version was tested for this review, though the documentation and the publisher say that the 8086 version appears identical to the user. The 8080/Z80 package includes the compiler, an assembler, a disassembler, and a screen editor. No linker was supplied, because separate compilation of program modules was not supported at review time. The linker and other enhancements are to be supplied with a release due in September, 1982.

The portion of Ada that is implemented is the basic core of the language. This includes all of the primitive datatypes of Ada, although there are a number of limitations and differences from the Ada standard in the implementation. Typical of the limititations are that an enumeration type is limited to 255 members, and enumeration literals may not be overloaded (i.e., the same value may not be used in two enumeration types, as it can be in Ada). Another limitation is that real types (floating and fixed point) are presently not implemented, although they are covered in the documentation and are promised for another release at a later date.

The primitive types that are allowed are constants (both object and number constants), enumeration types, booleans, characters, integers, bytes (not Ada standard), floating point (not yet implemented), fixed point (not yet implemented), arrays, strings (not per the Ada standard), records, variant records (not per the Ada standard), and access types. Fixed-point types (when implemented) will be in BCD so that rounding errors will not occur in business programs, and floating-point types (when implemented) will be binary with maximum precision of either 6.5 or 12 (long-float) digits.

Arrays may only have one dimension, but in my opinion, this does not present a handicap, since the elements of the array may be of any datatype, including another array. So, to obtain multidimensional arrays, they are declared as arrays of arrays. Each datatype has attributes assigned to it. While these are generally a subset of those in Ada, they are the most generally used ones (i.e., T'Size, T'First, T'Last, T'Pos, T'Val, T'Pred, and T'Succ), and can be used as arguments in most statement types. In addition user-defined types, derived types, and subtypes are supported.

Janus is a strongly typed language and explicit conversion between types is required. A derived type is considered to be different from its parent. In general, any numeric types or subtypes may be interconverted (subject to range constraints), and types derived from a common parent may also be interconverted. In this respect Janus is slightly less stringent than Ada, which requires that the parent be one of the types.

Basic features

Janus supports the full set of operators of Ada with the exception of slices and aggregates, but including the short-circuit forms of the logical operators. Allocators for runtime allocation and de-allocation of storage on the Heap are supported, except that initialization of the allocated object is not supported. Assignment statements are obviously

---

Harvey Fishman, 1408 East 49th St., Brooklyn, NY 11234.

supported, but runtime type and range checking is done, and failure of a match between both sides of the assignment in either type or range will cause a runtime error (this is a plus).

Direct assignment between arrays is supported, provided that both arrays match exactly. No longer is it necessary to assign element by element. Assignments are also allowed to string types, provided that there is sufficient room in the target variable. Janus strings have a dynamic length associated with them (similar to Basic or type character varying in PL/I); this is a departure from the Ada standard, where the length of a string is fixed.

All five of the basic control structures of structured programming are supported (**sequence, if-then-else, do-while, do-until,** and **case**). All except **do-until** is done with a continuous loop and a programmer-defined test for the **exit** statement that terminates the loop. In addition, **for** loops are supported. Additional control statements provided are the **return** and the **goto.** Thus the major control constructs of Ada are all provided in Janus.

In addition, Janus provides a (non-Ada standard) **Asm** statement as a method of including in-line assembly language statements. The arguments of this statement are included in the code as machine language. Two other quasi-functions (Janus calls them attributes) are provided to increase the usefulness of the assembly language capability: These are X'Address, which returns the starting address of any package object or subprogram whose name is substituted for the X; and P'Location, which returns the current value of the location counter when the name of the package you are in is substituted for the P. This allows assembly language access to variables and subprograms written in Janus—an extremely useful feature, since separate compilation and linking, which would be the normal way of getting at assembly language routines, are not yet provided.

Procedure and function subprograms are both supported with parameters passed only by reference. This is not standard Ada practice, which requires that scalar parameters be passed by value and permits structured parameters to be passed either by reference or by value at the option of the implementer. The Janus manual, however, states that you should not depend upon the method of parameter passage, since it may change in future versions. The Ada standard says much the same thing, though only for structured parameters. Actual parameter association is only by position, and default actual parameters are not supported. Overloading of subprograms and operators is not supported. Packages are supported with the standard Ada construction of a package specification and a package body, but until separate compilation is provided, this is rather academic.

Visibility rules are similar to those of Ada, but not identical, mainly because not all of the Ada constructs have been implemented. They are basically similar to those of any block-structured language, except that variables declared in an outer block that are masked by variables of the same name in the inner block are accessible using the dot notation. **Use** clauses are also supported to enable direct access to variables in other packages, but this is not of much use until separate compilation is provided. Renaming declarations to give new names to entities is not supported. Tasking for parallel processing is not supported, and the only compilation units that are allowed are package specifications and package bodies. Again, this does not matter until separate compilation is implemented. Exception handling is not implemented. Generic program units and representation specifications are not implemented.

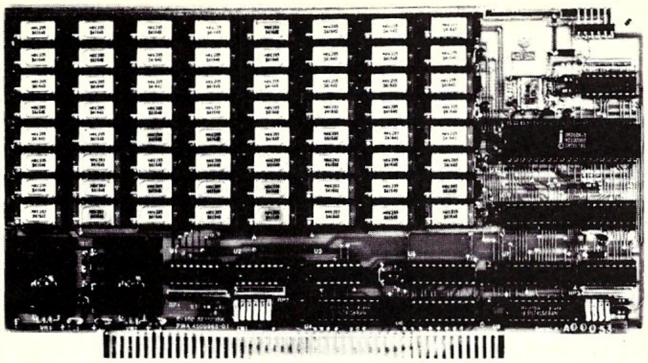
Handling I/O via libraries

I/O is provided in Janus via a library that may be included into source programs. No I/O is contained within the language defined by the Ada standard. Instead, a suggested set of I/O is defined by the standard and is to be included as packages with all validated compiler implementations. I/O is done this way since, in the intended use of Ada (embedded computers), most applications would have application-dependent I/O packages written specifically for the particular job to be done. However, to start off, the suggested I/O packages are provided and can be used at the discretion of the programmer. A similar suggested library is provided with Janus, too. Since separate compilation is not yet provided, this library must be used via an Include pragma (Ada's name for a compiler directive) that inserts the source of the I/O library into the source code for the compilation. This I/O library provides much of the flavor of the Ada standard suggested packages, but differs in details. All I/O is file-directed, the default file (if no file name is given in the command) being the console. In addition, the four standard CP/M logical devices (CON:, RDR:, PUN:, and LST:) are treated as files, and data may be transferred to or from them as appropriate.

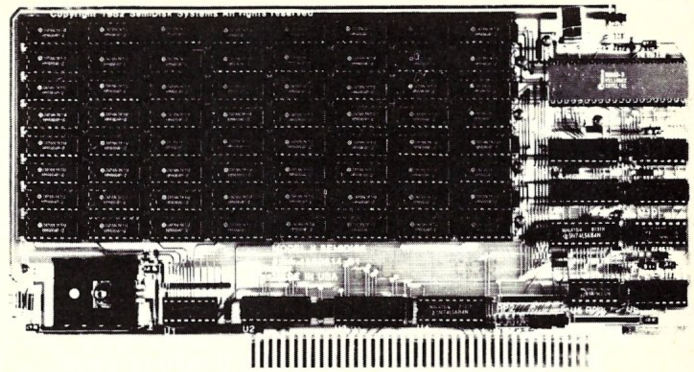
The following callable procedures and functions are provided in the I/O library. The procedure **Create** makes a new file of the desired name and leaves the file open for writing. If the file already exists or the directory is full, the procedure returns an error value in a global variable and it is the responsibility of the programmer to take further action. The procedure **Open** opens a file in any of three modes as defined by the procedure call: read, write, or read-write. If the file is not found, an error value is returned in the global variable. The procedure **Close** closes a file, updating the CP/M

In general, any numeric types or subtypes may be interconverted (subject to range restraints).

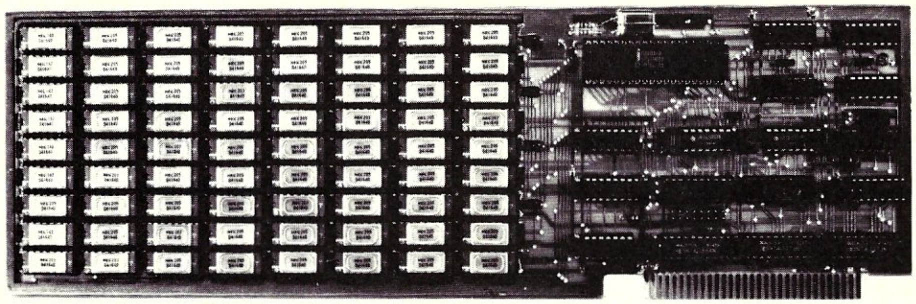
A FULL LINE OF SEMIDISKS



S-100



TRS 80 Model 2



IBM Personal Computer

Do you use your computer? Or does your computer "use" you? Face it, if you're using floppies, your time is being wasted. Because a floppy is an inefficient random access storage device. Each time the processor wants to transfer data, it has to wait an eternity for the disk to rotate and the head to move.

So what do you do? Get a SemiDisk, quick. It's a large capacity semiconductor memory board that is driven by software to operate like a disk drive. Without all the waiting. Do everything you'd do on a floppy or hard disk, with no modifications to your software or hardware. Two board sizes are available: 512K and 1 Megabyte. (the highest density microcomputer memory board in the world) And you can put up to 8 megabytes in a system by adding more storage boards.

What do you need to use it? Just an S-100 system with CP/M 2.2. Or a TRS-80 Model 2 system with CP/M 2.2. Or an IBM Personal Computer. That's it. No special processors, DMA, I/O, or disk controllers are required. Plug it in and run the installation program, and you're on your way. Fast! Even better, we supply full source code to the driver software, in case you'd like to do your own interfacing.

Best of all, the SemiDisk's price won't warp your wallet. Compare specs, cost/megabyte, storage capacity, and compatibility with the competition. You'll see that the SemiDisk is a disk emulator truly worthy of the name. SemiDisk has battery-backup capability, too.

Consider our limited warranty: A full year, covering all parts and labor. Consider our liberal 15 day return policy. Price? $1995 for 512K byte SemiDisk, $2995 for 1 Megabyte SemiDisk. Both from stock. $10.00 for manual. VISA, Mastercard, COD orders accepted. Dealer and OEM inquiries welcomed. (Specify system type and disk format when ordering.)

Someday, you'll get a SemiDisk.
Until then, you'll just have to . . . . wait.

SemiDisk SYSTEMS
P.O. Box GG
Beaverton, OR 97075
(503)-642-3100

Call (503)-646-5510 for CBBS®/NW, a Semi-Disk-equipped computer bulletin board.

CIRCLE 67 ON READER SERVICE CARD

directory and ensuring that the data is retrievable. This must be done explicitly by the programmer after writing to the file, as it is not done automatically upon program exit. The procedure **Delete** erases a file from the directory. The procedures **Read** and **Write** are provided to move binary data to and from a file. Functions End-of-File, Is-Open, and Name return, respectively, a boolean value showing whether the end of the file has been reached, a boolean value showing whether a particular file is open, and the external (CP/M) names of the file.

In addition to the binary file handling procedures, text handling procedures for ASCII files are included. The procedure **Get** moves ASCII data from a file. The data may be either integer or string data. The procedure **Put** moves ASCII data to the file. The data may be a string, an integer, or an enumeration type (including boolean). If no file is given as a parameter, the console is assumed. The procedure **New-Line** writes a <cr-lf> to the file or console, and the procedure **Skip-Line** causes the program to ignore all data up until the next end-of-line marker in the file. On input, a cr-lf, a cr alone, or an lf alone, will be interpreted as an end-of-line marker. The function End-of-Line returns a boolean value showing whether the file pointer is at an end-of-line in the file. The function Get-Line causes an entire line to be read from the named file, up to the end-of-line marker. When this function is used to input from the console, the normal CP/M text editing commands (^X, ^R, backspace, and rubout) can be used. They will not be recognized using Get calls.

Normal file I/O is sequential, but an additional library is provided to do limited random access file reads and writes using essentially CP/M 1.4 format. The object being read or written is limited to one full CP/M logical sector. Further, random access is limited to files whose size is one extent (16K) or less.

Other libraries

Additional libraries are provided (in source form) to perform string functions, runtime memory allocation control, and miscellaneous program control functions. The runtime memory allocation control is minimal, with functions to return the present value of the heap pointer and to free all memory that has been allocated since the allocation of the access variable passed to the function. In addition, a procedure is provided to return the amount of free memory available.

Another procedure, which implements the unchecked deallocation described in the Ada standard, is described in the Janus manual but is not yet implemented. The miscellaneous program control functions provided are Halt, which stops execution of the program and returns to the operating system; Err-Exit, which aborts the program and forces a runtime walkback (a very nice feature, described below) similar to that produced on an error; and Chain, which loads and executes a new program file.

Data passage to the new program is (barely) possible using a scheme similar to Fortran common. The best of these libraries is the string function library. This provides the functions Length, which returns the current length of a string; Remove, which allows you to delete a substring from within a string; and Insert, which allows you to insert a substring into a string. Also provided here are Extract, which enables you to take an arbitrary substring from the parent string, and Position, which returns the position of the first occurrence of a substring in a string. Other functions provided are conversions from character to string, string to integer, and integer to string.

What is included in Janus

As you can see, Ada is a large language indeed, and a fairly substantial subset has been implemented in Janus. The system is distributed on two single-density 8″ disks (although other formats are available). The compiler occupies 137K in six files, with three additional compiler support files occupying another 15K. The compiler requires a minimum of a 56K CP/M system to run, and obviously is a lot more comfortable with double-density 8″ disks, although (according to the publisher) it can be run on systems with less than 160K of disk space. The other programs in the package (assembler, disassembler, and text editor) require less space to run, but using less disk space restricts the size of the assembler symbol table and the editing buffer of the screen editor.

Using Janus

The Janus compiler can be both a joy and a pain to use. At present it is rather slow, but this is because the lack of separate compilation requires that the source of the libraries be included in the file to be compiled. This is done with the pragma, Include (<file-name>), which will load the file from the disk exactly like the similar statements in PL/I and BDS-C.

The size of these libraries is quite large, generally many times the size of a typical small program, so they greatly slow down the compiler. For instance, the file in Listing 1 is 78 lines, but the compiled file was 691 lines due to the inclusion of the I/O library. This took 6 minutes to compile. The Primes benchmark (based on the Sept. '81

Libraries are provided (in source form) to perform memory allocation and miscellaneous control functions. Normal file I/O is sequential, but a library is provided to do limited random access.

SoftwareBanc

661 MASSACHUSETTS AVENUE, ARLINGTON, MA 02174

THE dBASE II SPECIALISTS

dBASE II™

WE WROTE THE *ONLY* BOOK

dBASE II™ USER'S GUIDE $29
FREE WITH dBASE II™!

WE OFFER THE *LATEST* ACCESSORY PRODUCTS

dUTIL	$ 69	dBASE II™ UTILITY PACKAGE
QUICKCODE	$229	dBASE II™ PROGRAM GENERATOR
NEW! ABSTAT	$379	STATISTICS PACKAGE FOR dBASE II™ FILES

WE STOCK THE *WIDEST* SELECTION OF dBASE II™

dBASE II™ WITH *FREE* dBASE II™ USERS GUIDE
CALL FOR PRICE

8" SINGLE DENSITY, TRS-80 MODEL II
5¼" APPLE II/III, HP-125, NORTHSTAR, SUPERBRAIN, TELEVIDEO, VECTOR GRAPHIC, XEROX, ZENITH Z-89
IBM PC DOS 1.1

WE TEACH *ALL* THE CLASSES

SPONSORED BY SOFTWAREBANC SEMINARS, INC.

DAY 1	INTRODUCTION TO dBASE II™	9AM - 5PM	$100
DAY 2	ADVANCED dBASE II™ TECHNIQUES	9AM - 5PM	$100

BOSTON
NOV. 6 INTRO
NOV. 7 ADVANCED

CHICAGO
NOV. 20 INTRO
NOV. 21 ADVANCED

MIAMI
DEC. 28 INTRO
DEC. 29 ADVANCED

ORDER **1-800-451-2502**
(617) 641-1241 IN MASS.

TECHNICAL SUPPORT (617) 641-1235

Payment may be made by Mastercard, Visa, check, money order, wire transfers. Mass. residents please add 5% sales tax. Add $5.00 for shipping and handling. Overseas orders add additional $10.00. Prices subject to change without notice. All items subject to availability. ™ - Mfg. Trademark

CIRCLE 35 ON READER SERVICE CARD

Byte benchmark program) was 30 lines and did not include the I/O library, and so only took a minute and a quarter to compile.

The joy of using this compiler is that it gives *really good* error diagnostic messages. They are understandable and there is a pointer to the exact place in the file where the error occurred. They even tell you which file the error was in if you have multiple files using the Include pragma. The compiler will try to correct minor mistakes, such as the omission of a required semicolon. When it does this, it will ask you if this is acceptable. You can accept the correction, and the compiler will continue, or else you can abort the compilation and make the correction in the source file. These features, unfortunately, lead to the pain (hopefully, temporary) of using the compiler.

At present, there are a number of unimplemented features in the language that are promised for the future. Alas, the compiler does not know this, and although it will tell you when you use a deliberately unimplemented feature, when it runs into one of the temporary problems it goes off into a never-never land of wrong error messages. This can be very frustrating until you figure out what is going on.

The most beautiful feature of this compiler is the error-checking code that it generates. When a runtime error occurs, the error checkers included in the object code generate a walkback through the nested levels of the program and show you where the error occurred. A walkback is a list of the called modules within the program in calling order showing you just where in the program the error occurred and how you got there. Other languages (such as Digital Research's PL/I) also generate this walkback, but they do it in terms of the addresses on the stack. Janus does it in terms of the line numbers and the module names in the program source. Here is a typical runtime error message:

```
** Subscript or Subrange Out of Bounds - Pos of Error Value = 36
On Line Number 13 In TRY14.PROC3
Called from line number 19 in TRY14.PROC2
Called from line number 25 in TRY14.PROC1
Called from line number 34 in TRY14
```

Wow, does that ever make debugging easier! Of course this much error checking makes the code larger, but with the decline in memory prices, that is no longer such a major consideration. If you really feel that you do not want the overhead of the debugging code, there is a pragma that will prevent its generation. However, I feel that it would generally be foolish to defeat it. All in all, this is probably the finest compiler I have seen for any microcomputer language, and when it is sorted out (hopefully with the September 1982 release), it should be *super*.

The code generated is not particularly fast, but that is probably to be expected in a language that does as much runtime checking as Janus. Direct comparisons were made with PL/I and BDS-C in two completely different types of programs. The first benchmark used was the Sieve of Eratosthenes Prime Number Program from the September 1981 issue of *Byte*. The source for PL/I and BDS-C appeared in the *Byte* article, and the source for Janus was distributed as a sample program in the package. The times for these runs are given in Table 1. This program does no disk I/O, so I wrote a benchmark that is heavily I/O bound. This program is shown in Listings 1, 2, and 3 for Janus, PL/I, and BDS-C respectively. It replaces spaces in an ASCII file with 8-space tabs wherever possible. It was run on a 104K assembly listing that had no tabs in the file (all white space was ASCII spaces). The times for these runs are shown in Table 2.

Table 1. Primes benchmark

Janus	18.3 seconds
PL/I	9.7 seconds
BDS-C	26.5 seconds

Table 2. Tabbing benchmark

Janus (128-byte buffer)	690 seconds
PL/I (128-byte buffer)	668 seconds
PL/I (1024-byte buffer)	348 seconds
BDS-C (1024-byte buffer)	122 seconds

Janus is very slow because it presently uses 128-byte buffers for disk I/O, and a heavily I/O-bound program like this causes a lot of thrashing. I ran the PL/I version of the program with both 128- and 1024-byte I/O buffers, and when 128-byte buffers were used, it took almost exactly the same time as Janus. BDS-C was run with 1024-byte buffers only, because it is extremely difficult to change buffer size there.

The Janus assembler

The assembler provided with the package is a relocating assembler that uses a combination of the TDL/CDL Z80 superset of Intel Mnemonics with the Intel pseudo-ops. No information is given about the relocation format other than that it will be compatible with the Janus linker when that is released. The main purpose seems to be to provide a means of linking assembly language modules to the Janus compiled modules. About the only thing that seems remarkable about it is the fact that it makes no restriction on the length of the significant part of user-defined symbols.

The Janus compiler can be both a pain and a joy to use. At present it is rather slow. . . . The joy of using this compiler is that it gives really good diagnostic messages.

The disassembler supplied seems to be only useful for disassembling Janus object code. It can be extremely useful for debugging purposes, as Janus does not generate a symbol file for use with SID. The listing generated by the disassembler has the assembly language code interspersed through a listing of the Janus source file, relating the executable code to the source. The file cannot be reassembled without extensive editing, because it is designed to be human-readable, and does a fairly good job of that. It is useful for figuring out how Janus programs work, and by using it with DDT, there are no unusual problems in debugging compiled code. The text editor supplied with the package is in rudimentary form, and the documentation suggests that if you have a good screen editor, you are probably better off sticking with it. I took the suggestion.

Janus documentation

The documentation supplied with Janus is fairly good, when you realize what it is meant to be. It is not a course in the language, but merely a document to show how to use the Janus system, and how it differs from the Ada standard. The Janus manual also tends to restate the words of the Ada standard in a more readable form, which is probably a godsend. The publishers suggest that you obtain a textbook on Ada and use that as a basis for learning the language, and that is probably the best idea for any language. The Janus documentation is organized with exactly the same section headings as in the Ada standard, so that it is easy to determine just how Janus differs from standard Ada. Ada is such a large language that it is difficult to conceive of anyone programming in it without a copy of the standard at his side (in fact, this is one of the major complaints about the language). The Janus manual should be one more book at your side while programming. The main problem with the documentation is that it does not always agree with what the compiler actually does,

Listing 1 — Tab program in Ada

```
package body entab is            -- replaces spaces with tabs

pragma include ("io.lib");
pragma include ("string.lib");

tab : constant := 8;
a : integer;
inname, outname : string;
infile, outfile : file;

procedure tabify is
c : character;
position, space_count : integer := Ø;

begin
  while not end_of_file (infile) loop
    get (infile, c);             -- get next character
    if c = ' ' then              -- if it's a space
      position := position + 1;
      space_count := space_count + 1;
      if (position mod tab) = Ø then
        put (outfile, character'val(9));
        space_count := Ø;
      end if;

                                 -- if it's a tab
    elsif c = character'val(9) then
      position := position + 1;
      space_count := Ø;
      put (outfile, c);
      while (position mod tab) /= Ø loop
        position := position + 1;
      end loop;

                                 -- if it's a line feed
    elsif c = character'val(1Ø) then
      put (outfile, c);

                                 -- if it's a carriage return
    elsif c = character'val(13) then
      position := Ø;
      space_count := Ø;
      put (outfile, c);

                                 -- if it's anything else
    else
      position := position + 1;
      while space_count /= Ø loop
        put (outfile, ' ');
        space_count := space_count - 1;
      end loop;
      put (outfile, c);
    end if;
  end loop;
end tabify;
```

but I have an early release and hopefully this will be cleared up.

The documentation for the assembler is again not a course but a reference manual, and the publishers suggest here also that you obtain a text to learn assembly language programming if required. As a reference manual the documentation supplied is very good. The disassembler program is so simple to use that very little documentation is required. The manual tells you how to run it and how to edit the output file into a form that can be assembled if you so desire. This is more than adequate. Finally, the documentation for the text editor seems to be fairly complete, but as I did not venture to use the editor, this is just a first impression. The people at RR Software were extremely helpful and cooperative when I called them (without identifying myself as a reviewer).

Conclusions

All told, I think Janus is a very worthwhile implementation of a major subset of the Ada language, and while there are still quite a few rough edges, hopefully they will be straightened out with the next release, which should be available by the time you read this. The compiler breaks new ground in the microcomputer field with its excellent runtime error-checking code and its excellent compiler error messages. The package is fairly complete (or will be when separate compilation and real data types are released), with a compatible assembler for assembly language routines when required, and the disassembler for additional debugging aid. Janus is available from RR Software, P.O. Box 1512 Madison, WI 53748. The latest price is $300 for the 8080/Z80 version and $400 for the 8086 version.

Note: *All times shown here were obtained from runs made on a 61K system with a 6-MHz Z80 processor, using double-density/single-sided disks with an 8-millisecond track-to-track step time and 1K sectors with 2K blocks.*

```
begin
    init_io;
    put ("File to entab? ");          -- get input file name
    get_line (standard_input, inname);
    a := position (".", inname);      -- construct output file name
    if a = 0 then
        a := length (inname) + 1;
    end if;
    outname := extract (inname, 1, a - 1) & ".tab";
    open (infile, inname, 1);         -- open input file
    if ioresult /= 0 then
        new_line;
        put ("File not found!");
        new_line;
        err_exit;
    end if;
    create (outfile, outname);        -- make output file
    if ioresult /= 0 then
        new_line;
        put ("Unable to open output file!");
        new_line;
        err_exit;
    end if;
    tabify;                           -- process file
    close (outfile);                  -- close output file
end entab;
```

Listing 2 — Tab program in PL/I

```
entab:                              /* replaces spaces with tabs */
    proc    options (main);

    %replace
        tab by  8;

    dcl (inname, outname) character(20) varying,
        (infile, outfile) file,
        a binary;

    tabify:
        proc;

        dcl
            (c, temp)    character(1),
            (position, space_count) binary,
            eof bit(1);

            eof = '0'b;
            position = 0;
            space_count = 0;
            on endfile(infile)
                eof = '1'b;
            do while (^eof);
                                    /* get next character *
                read file (infile) into (c);
                                    /* if it's a space *
                if c = ' ' then do;
                    position = position + 1;
                    space_count = space_count + 1;
                    if mod (position, tab) = 0 then do;
```

```
                temp = ascii(9);
                write file (outfile) from (temp);
                space_count = 0;
            end;
        end;
                              /* if it's a tab        */
        else if c = ascii(9) then do;
            position = position + 1;
            space_count = 0;
            write file (outfile) from (c);
            do while (mod (position, tab) ^= 0);
                position = position + 1;
            end;
        end;
                              /* if it's a line feed    */
        else if c = ascii(10) then
            write file (outfile) from (c);
                              /* if it's a carriage return   */
        else if c = ascii(13) then do;
            position = 0;
            space_count = 0;
            write file (outfile) from (c);
        end;
        else do;              /* if it's anything else   */
            position = position + 1;
            temp = ' ';
            do while (space_count ^= 0);
                write file (outfile) from (temp);
                space_count = space_count - 1;
            end;
            write file (outfile) from (c);
        end;
    end;
end tabify;

                              /* get input file name   */
put list ('File to entab? ');
get list (inname);
a = index (inname, '.');      /* construct output file name   */
if a = 0 then
    a = length (inname) + 1;
outname = substr (inname, 1, a - 1) || '.tab';
                              /* open input file       */
on undefinedfile (infile) begin;
    put list ('File not found!');
    stop;
end;
open file (infile) sequential env(b(1024)) title (inname);
                              /* make output file      */
on undefinedfile (outfile) begin;
    put list ('Unable to open output file!');
    stop;
end;
open file (outfile) sequential output env (b(1024)) title (outname);
call tabify();                /* process file          */
close file (outfile);         /* close output file     */

end entab;
```

Listing 3 — Tab program in C

```
#include <bdscio.h>

#define TAB     8

main()          /* replaces spaces with tabs */

{
    char    inname[20], outname[20], inbuf[BUFSIZ], outbuf[BUFSIZ];
    int     i;

    i = 0;
    printf ("File to entab? ");          /* get input file name */
    scanf ("%s", inname);
                                         /* construct output file name
    while (inname[i] != '.' && inname[i] != '\0')
        outname[i] = inname[i++];
    outname[i] = '\0';
    strcat (outname, ".tab");
    if (fopen(inname, inbuf) == -1) {    /* open input file */
        printf ("File not found!");
        bdos(0);
    }
                                         /* make output file */
    if (fcreat(outname, outbuf) == -1) {
        printf ("Unable to open output file!");
        bdos(0);
    }
    tabify (inbuf, outbuf);              /* process file      */
    fflush (outbuf);                     /* close output file */
    fclose (outbuf);
}

tabify(inbuf, outbuf)
char    inbuf[], outbuf[];

{
    char    c;
    int     position, space_count;

    position = space_count = 0;
                                         /* get next character */
    while ((c = getc(inbuf)) != EOF && c != CPMEOF) {
        if (c == ' ') {                  /* if it's a space */
            position++;
            space_count++;
            if ((position % TAB) == 0) {
                putc('\t', outbuf);
                space_count = 0;
            }
        }
        else if (c == '\t') {            /* if it's a tab */
```

Janus— continued . . .

```
        position++;
        space_count = 0;
        putc(c, outbuf);
        while ((position & TAB) != 0)
            ++position;
    }
    else if (c == '\012')          /* if it's a line feed */
        putc(c, outbuf);
    else if (c == '\015') {        /* if it's a carriage return*/
        position = space_count = 0;
        putc(c, outbuf);
    }
    else {                         /* if it's anything else*/
        position++;
        while (space_count != 0) {
            putc(' ', outbuf);
            space_count--;
        }
        putc(c, outbuf);
    }
}
putc(CPMEOF, outbuf);
}
```

SOFTWARE REVIEW

Two More C Compilers

A comparative review of the Aztec C II and C/80 compiler

by David A. Gewirtz

In my previous articles on the C Programming Language (*Microsystems*, Nov/Dec 1981 and Jan/Feb 1982), I gave a short general description of the C language, and then followed with reviews of four rather different implementations of the language for CP/M. Each implementation appealed to a different audience, with Small-C on the low end, BDS-C serving a mid-range, and Whitesmiths serving a high end. Out by itself was Tiny-C, which is more of a training system than a production compiler.

This article can be seen as sort of an informal continuation of those reviews. All of the implementations examined here fit into the same general mid-range as BDS-C.

The Aztec C II Compiler

The Aztec C Compiler was developed and is being marketed by Manx Software Systems of Shrewsbury, New Jersey. The Compiler is actually available in a number of different versions. The original is Aztec C. This is the actual C Compiler and some utilities. The Aztec C II Compiler is simply the Aztec C Compiler with floating-point features.

The C II Compiler also comes in two different versions, one for the 8080 and one for the Z80. Both are supplied with the distribution package. They also make versions of the compiler for CP/M on the Apple, Apple DOS, HDOS on the Heath and Zenith machines, an 8086 version for CP/M-86, and a C for the IBM-PC. The version of Aztec C II reviewed here is Aztec C II, Version 1.02, for the Z80. According to the documentation, the compiler requires a true 56K CP/M system and CP/M version 2.2 to work properly.

Aztec C II is a single-pass compiler that generates relocatable assembly language code, acceptable to either Microsoft's M80 macro assembler, or Manx's own AS relocating assembler. The choice of assemblers can be dictated by giving a '-M' option to the compiler. When the Manx AS assembler is used, it generates object code acceptable to a linkage editor called LN, which links all external library routines to the mainline program and generates an executable 'COM' file.

As stated previously, the Aztec C II package comes with an 8080 and Z80 version of the compiler and runtime package, the relocating assembler, and the linker. It also comes with the source code for the support routines needed by the compiled programs, as well as sources for all of the library functions.

The documentation is relatively clear and easy to understand. However, for the sake of clarity, I feel that it needs some organizational revisions. There are times, such as when it explains the allocation of table sizes at compilation, that the descriptions and examples could be in more detail. The manual itself is approximately 50 pages long and bound in a three-ring binder.

Significantly absent from the Aztec C package are development utilities, such as a C library manager. As programs get larger and more time is spent developing software with one package, a utility to manage the object code routines becomes critically necessary. There is a sort of librarian available with the Microsoft M80 macro assembler, and I suspect that if library management is necessary, the M80 option would have to be chosen.

The Aztec C II Compiler comes complete with nearly all of the standard C features that might seem necessary. It has all of the standard C language features, including floats, longs, register variables, and statics, with the exception of bit fields. The documentation seems to imply that Aztec C II is a full implementation, but does not itemize all of those features in a coherent fashion. The language seems to have the preprocessor directives '#include' and '#define', but makes no definitive reference to them or any others in the documentation. It should be noted, however, that the manual does state that it uses the Kernighan and Ritchie book as the Aztec C II specification, and does in fact note those things that do not comply. In summary, Aztec C II is a nearly complete implementation of C.

The compiler supports most of the library routines commonly associated with C, including formatted printing and buffered I/O. It also supports a form of I/O redirection, as well.

A program written for the Aztec C II Compiler is compiled in three steps. The first is compilation:

```
A>CII EXMPL.C
```

This has options for the sizes of various tables, as well as whether the C statements should be included as comments in the assembly code. The second phase is assembly with the AS assembler:

```
A>AS EXMPL.ASM
```

Both the compilation and assembly phases are fast and easy to use.

The final phase is the linkage phase. This is

David A. Gewirtz, 1 Captain Drive #457, Emeryville, CA 94608

specified as:

```
A>LN EXMPL.O -L LIBC.LIB
```

The '-L' option specifies that LIBC.LIB is a library file. Since LIBC.LIB must be included as a library file in every linkage, it would seem that the authors of the system might have included it in a default search of libraries. The linkage phase is also rather slow, and can be annoying to wait for.

One final note of interest is the use of register variables. The register declaration specifies that those items declared as register should be kept in the processor's registers whenever possible. This feature can, at many times, speed up processing greatly. As an example, I performed two of the performance evaluation tests from my January/February 1982 article and found a savings of 14 to 48 percent. Using PE2, simple integer calculation, I replaced the statement:

```
        int i, j, k, l;                 (1)
```

with:

```
        register int i, j, k, l;        (2)
```

Using statement (1) in the code, the program ran for an average 10.7 seconds. Using (2), it ran for 9.3 seconds. This is not as impressive as the improvement found in PE4, pointer operations. I replaced the statements:

```
        char arry[128], *ptr;           (1)
        int i;
```

with the statements:

```
        char arry[128];
        register char *ptr;             (2)
        register int i;
```

The program using the declarations in (1) took 314.3 seconds to execute, while the code using (2) took only 162.8 seconds, for a savings of 48 percent. I did the same kind of substitution with the Sieve program from the September 1981 *Byte* magazine. Without using register variables, it took 33.0 seconds to execute; using them, the program executed in 22.9 seconds, yielding a savings of 31 percent.

In actual terms of speed optimization, the above percentages are critical to consider. It is important to realize the restrictions of the machine, however. Remember that any processor has only a small number of registers, and overloading them or using them inefficiently can cause more problems than it cures.

Overall, I am reasonably pleased with the Aztec C II Compiler. I would like to see some changes made to make the compiler and documentation easier to use, as well as the addition of some program development utilities. In other words, it can be used nicely now, but could use some minor work.

Software Toolworks C/80 2.0

The Software Toolworks C/80 C Compiler evolved originally from Ron Cain's public domain Small C Compiler. Cain's Small C was a minimal compiler that served as an inexpensive bootstrap into the world of C programming. Since its release, at least four other commercial C compilers have been released, all derived from Small C.

C/80 is rooted in the Small C Compiler, but there the similarities end. C/80 has many of the features of a full C compiler and executes much faster than Small C. The only similarity to Small C is the price. C/80 is $49—quite a bit less than any other comparable compiler.

The C/80 2.0 compiler comes in versions for CP/M and HDOS, and can run on either an 8080 or Z80 processor with a minimum of 48K memory.*

The C/80 compiler takes C source code and generates an assembly language source code file. This file can then be assembled into object code by another AS assembler, this one from the Software Toolworks. An option allows code to be generated for Microsoft's M80 macro assembler.

In the case of this compiler, it is almost necessary to have M80 if any production work is necessary. Using AS as an assembly and linkage agent, all external functions must be referenced in the mainline source code as '#include' files, and be recompiled and reassembled with every program.† M80 uses relocatable elements and a linker to allow external routines to be referenced as object files. It also allows the C code to be intermixed with object code from other compilers.

The C/80 package comes with a 35-page unbound manual and a distribution disk, as well as some literature about the company's other products. The manual is well thought out and clear. It includes a table of contents on the front page, as well as (are you ready for this?) an *index*. In the manual are sections detailing the contents of the distribution disk, an example compilation, a summary of the C/80 language, machine and implementation dependencies, the runtime and I/O library, as well as sections on using C/80 with M80, and some other special interest sections. All in all, the manual is well done, informative, and to the point.

The distribution disk contains the executable compiler, a source code listing of the C/80 I/O library, various high-level source routines such as PRINTF, the AS absolute assembler, as well as two wonderful utilities called CTRACE and CPROF, used for runtime trace and execution profile.

The trace and execution profile utilities are routines that can be hooked into the source code of any given program to aid the programmer in debugging and testing. CTRACE performs a trace of all functions executed by the program during a

**See Software Toolworks' comment No. 1.*

†See Software Toolworks' comment No. 2.

The Aztec C II compiler can be used nicely now, but needs minor work and a few more utilities.

run. When invoked, it prints out a message stating that it has entered a library function, and upon exiting, specifies the name of the function and the value returned. None of the functioning of the actual program is impaired, and all input and output is done in between trace messages. The CPROF utility works in much the same way as CTRACE, but displays a listing at the end of the total times all functions were executed.

These utilities give the programmer an idea of what can be done with a compiler, but are only in the infant stages. There is little or no control of the tracing facilities other than invoking them. The trace and profile utilities cannot be run at the same time; after using one, the program must be recompiled to get the other to work. The ability to use these relies almost entirely on M80, because otherwise the programmer is forced to rename the utilities to the name searched for by the compiler. These utilities are a start. They are not on a par with debugging aids found on mainframes or minis, but are certainly more than what is available on other C compilers for CP/M.

The C/80 compiler supports most of the standard C features with the exception of float, double, longs, typedef, and bitfields. All other functions, storage classes, datatypes, and operators are supported. In the list of operators listed in the documentation, '%=' was missing, but Walt Bilofsky of The Software Toolworks has informed me that it does indeed exist, and its omission was a typographical error.

In the performance evaluation tests I ran, I found that code generated by C/80 was typically much faster than that of many of the other compilers, and took a relatively short time to compile and assemble/load using the AS assembler. I did run tests using register variables as well, but did not find anywhere near the significant savings found in the Aztec C Compiler. The use of the register declaration caused a savings of zero (PE2) to 9 percent in the Sieve program.

The compiler comes with a small set of library functions that can be called by the user. Many of the more advanced functions can easily be built out of these, but it seems that some of the more common functions, such as **gets,** could have been implemented or included.

Also significantly absent from the list of callable functions were functions allowing access to the native operating system, CP/M. No facility was provided to allow access to BDOS and BIOS calls. In our conversation, Bilofsky stated that he felt that they could be implemented rather easily, and also that they would limit portability between machines. I tend to feel that access to the native operating system is necessary when the set of library functions is limited, and the package is used extensively for software directly related to that operating system. In the interests of mankind as a whole, and programmers of C/80 as well, I have included the code to perform BDOS calls from C/80. The calling sequence is:

bdos(c,de)

in which **c** contains the BDOS function number, and **de** the argument. The code, using inline assembly source, is the following:

```
bdos(c,de)
int c;
int de;
/*
     Stack arrives as:
     bottom-    C
                DE
                Return address of function call
*/
{
#asm
          POP  B     ; RETURN ADDRESS
          POP  D     ; CONTAINS ARGUMENT IN DE
          POP  H     ; CONTAINS FUNCTION NUMBER IN L

          PUSH H     ; THE STACK MUST BE IDENTICAL
          PUSH D     ; TO WHAT IT WAS WHEN CALLED
          PUSH B
          MVI  C,L   ; PUT FUNCTION NUMBER IN L
          JMP  0005 ;BDOS ENTRY POINT

          ; DE ALREADY IS SET TO DE
          ; BDOS WILL EXECUTE A RETURN, AND C/80
          ; WILL GENERATE A RET FROM THE FUNCTION
#endasm
}
```

You will note that the actual assembly code is in upper case. This is because the AS assembler gets rather upset about code in lower case. It probably would make sense for the C/80 compiler to translate to upper case inside #asm/#endasm, or for AS to understand lower case.

For those who have terminals that support only upper case, C/80 has a preprocessor option called '#UPPER'. This allows the programmer to use symbol combinations to represent upper- and lower-case characters.

C/80 has some minor bugs/problems that are currently being examined. One such problem allows C/80 to write only an even number of records to a file. Another causes global arrays to occupy physical space in the generated 'COM' file. Both of these are important, but not critical to the overall value of the compiler.

In conclusion, I must say that I am impressed. C/80 2.0 has most of the features necessary in a production compiler. It also generates fast code, and does not take too long to compile and link programs. It has some utilities that can greatly help in the debugging process, as well. For $49, it is something special. I think that if it were priced quite a bit higher, I would demand some of the features mentioned above, but few other compilers are as good, even for a much higher price.

C/80 has most of the features necessary in a production compiler . . . The manual is well thought out and clear. For $49 this package something special.

COMMENTS FROM THE SOFTWARE TOOLWORKS:

1. C/80 is available on 8″ single-density diskette for CP/M, and on 5¼″ diskette for the Osborne computer (CP/M) and the Health/Zenith computer (HDOS).

   A version for the Exidy Sorcerer (CP/M) is available from Triangle Systems, 1690 West Lane Ave., Columbus, OH 43221.

   A version to run under CP/M-86 is in preparation.

2. Separate modules may also be compiled individually and incorporated using the XTEXT pseudo-operation of the assembler.

Performance evaluations

I performed the same set of performance evaluations on Aztec C II and C/80 as were performed in the previous articles on the other compilers reviewed. Additionally, since a fine Sieve benchmark was written up in the September 1981 *Byte* magazine, I included tests of the compilers using it. The sieve test was simply reflected on the execution speed tests, and did not affect the averages for compilation time and program size.

In addition, I tested the two compilers with register variables using tests PE2, PE4, and the Sieve. The timings for those are noted in parentheses after the normal timings.

I was asked (pleaded, begged) by the folks who wrote Aztec C II to include some commentary and benchmarks of their floating-point facilities against other compilers supporting floating point, specifically Whitesmiths. I will mention that Aztec C II does support a complete set of floating-point and long operations. However, I am not an expert in floating-point systems and do not have any papers immediately available that specify floating-point benchmarks. If I receive a few papers detailing a few different floating-point benchmarks, I will run them and include the results in a short article at some time in the future, benchmarking Aztec against any other floating-point C systems.

Conclusions

Overall, I was impressed by both products, and since they have an appeal to the mid-range market, I feel a need to compare them to the BD Software C Compiler. In terms of the completeness of the actual compiler, both packages exceed BDS. In terms of ease of use, and overall programming environment, BDS exceeds both of these packages. The BDS implementation represents an approach much more suitable to production work, while the C/80 is a more complete compiler generating faster code, and Aztec is nearly complete with the

Performance Chart

Name: **Aztec C II**
Price: $195 plus shipping
From: *Manx Software Systems*
Box 55
Shrewsbury, NJ 07701
(201) 780-4004

Name: **C/80 2.0**
Price: $49.95 plus shipping
From: *Software Toolworks*
14478 Glorietta Drive
Sherman Oaks, CA 91423
(213) 986-4885

Item compared	AZTEC C II		C/80 2.0	
Average program size (recs)	62		26	
(kilobytes)	8		4	
Timings (in seconds):				
Average compilation and linkage	75.73		68.5	
Compilation	11.8		26.0	
Assembly	10.43		42.5	
Linkage	55.3		NA	
Benchmark tests:				
PE1—Simple counting loop	28.4		21.4	
PE2—Simple count and integer calculation	10.7	(9.3)	6.2	(6.2)
PE3—Conditional evaluation	8.6		7.3	
PE4—Indirection (pointer) operations	314.3	(162.8)	228.3	(226.3)
PE5—Simple function calling (no arguments)	101.2		27.0	
PE6—Function calling with argument passing	138.4		65.9	
Sieve	33.0	(22.9)	27.1	(24.9)

(Numbers in parentheses represent timing, using register declarations.)

potential of executing very fast code as well.

It is, however, still possible to identify ranges of appeal for these three compilers. C/80 should appeal to the new C programmer, or one who needs speed, without floating point. Aztec C II should appeal to the programmer wishing to design programs portable to other environments, such as Unix. BDS is the most comfortable to use, but slightly slower and containing fewer datatypes. All three are excellent for writing utility and basic system routines, but if this is all you wish to do, I would recommend C/80 as an inexpensive and powerful compiler. If you need a full working C language, I would choose Aztec C II over Whitesmiths, both in ease of use, portability, and cost. Generally, for overall work, I would still choose BDS, for its ease in linking programs together.

David Gewirtz has a B.S. honors degree in Computer Science from Worcester Polytechnic Institute and is currently a Ph.D. student at the University of California at Berkeley. In addition to writing truly fascinating articles about computers and languages, he has received a Sigma XI Research Award in Engineering for a new structurally extensible language and command environment called the Hope System, and has written philosophical papers about computer's rights. In terms of reality, he has a Jeep (at least at the time of writing; hopefully at the time of printing) and is attempting to make noise (someday, maybe even music!) with a guitar.

NorthStar™ Topics

A Virtual Disk Facility for North Star DOS

by D. J. Anderson

Recently, as I watched my mini-floppies churning away while assembling a large program, it occurred to me that I could speed up execution considerably if I could dedicate a part of my RAM memory to emulating a disk.

Ideally, such an emulation would be transparent to the user and could be invoked or disconnected at will. Whenever a disk access occurred, the emulator would check to see if it was for a real drive or the emulated drive. If real, it would pass the request to the operating system for processing. If for the emulated disk, it would perform the required function, but use a RAM buffer instead of the real disk. The advantage would be in access time: There would be no rotational delay due to a physically spinning disk, no drive start-up time, no track seek time, and no head settling time. The disadvantage would be that RAM space would be dedicated to the emulated disk.

North Star DOS is ideal for this concept, as it is well documented, and all disk processing is done through one routine (DCOM), which is in a known, fixed location.

In order to emulate a disk, the routine must dedicate part of the RAM buffer to the disk directory. Standard North Star DOS uses four sectors of 512 bytes (double density) or 256 bytes (single density) to store the directory. Each entry in the directory requires 16 bytes, thus giving 32 (double-density) or 16 (single-density) file references in each sector. Since the amount of RAM space is severely limited in most computers, I felt it sufficient to provide only one directory sector instead of all four. Thus the routine must bypass all references to sectors 1–3, the second through fourth directory sectors.

The program shown in the accompanying listing consists of an initialization routine and a processing routine. When first invoked, the program initializes the directory, saves the current DCOM vector, and then alters it to point to our processing routine, called FASTRAM. It then calculates the size of the RAM buffer, displays the size on the screen, calculates the maximum sector address of our pseudo-disk, and returns to DOS.

Now, whenever disk activity occurs, the FASTRAM routine is called due to the altered DCOM vector. This routine first checks to see if we are accessing drive 4 (which does not exist in my system). If we are not, it merely jumps to the original DCOM routine to let it handle the disk activity. If it is drive 4, however, we must emulate the required activity.

Four activities need to be handled, and these are indicated by the value in register B, as described for DCOM in the North Star system manual. When B is negative, the disk is being initialized; when B is 0, we are writing one or more sectors; when B is 1, we are reading one or more sectors; and when B is 2, we are verifying that what was written matches what is in memory.

The only tricky function is the initialization: Register DE will point to a 512-byte buffer containing all blanks. We want our emulation program to initialize only the actual RAM buffer, and to ignore all writes beyond the end.

However, when normal writing, reading, or verifying occurs, we want to abort any function beyond the end of our RAM buffer, and display an error message.

References to the missing directory sectors must be ignored if a write is taking place; a read must return a blank directory sector; a verify must return a match condition.

The program as shown runs on North Star DOS, release 5.2, which has its origin at 0100 hex. It is set up for double density, but will run on single density if you change the number 512 to 256 wherever it occurs, and change the 32 directory entries to 16 in the FASTENTY routine.

The routines CRLF, MSG, MSGCR, and HEXOUT all exist in my resident BOOT PROM; if you do not have one that has similar routines, then merely omit the section of the initialization routine that displays the size of the RAM buffer. However, you will have to change the display of the error message at ERROR2 to use whatever routines you have. One possibility is merely to jump to the DOS hard disk error routine.

One other peculiarity of the assembler used to print the listing is that the mnemonic ASCC, used to create MSG1 and MSG2, compiles the specified characters with a carriage return at the end.

The program should be compiled for the top of memory, leaving sufficient room for the RAM buffer. In my system, I have 48K of continuous memory, so I compiled the program for address A000 hex; the program itself takes less than 500 bytes, thus leaving 7.5K (15 sectors or 30 blocks) for the RAM buffer. A 30-block disk drive is not large, but it is sufficient to demonstrate the speed of emulation. If you have two or more bank-

D. J. Anderson, 755 Southmore Drive West,
Ottawa, Ontario, Canada K1V 6Z9

switched memory boards, then you can vastly increase the size of the RAM buffer by putting it in the second board. You would then have to make some minor modifications to this program to switch the boards in and out of memory space to transfer the data. In this case, the program must sit in unswitched memory (such as a PROM), so it can properly transfer the data between the two boards.

To use the program, you merely invoke it as any other program, using the GO command. It attaches itself to the DOS, initializes its pseudo directory, displays the size of the pseudo disk, and returns to DOS. Thereafter, you can use any DOS commands, and run any program, including Basic, referencing drive 4 as if it were really there. All actions will take place at lightning speed. Remember, though, that this disk evaporates when the power goes off, so copy anything on it back to a real disk.

Note that when you use Basic, you must use the MEMSET command to lower the top of memory limit to leave space for the program and the buffer. If you use a program that scans memory for the physical end, such as Allen Ashley's COMSTAR package, you must modify it so it will not clobber this program.

To disconnect the program, all you have to do is re-boot the DOS.

If you have bank-switched memory, and you want to always have the pseudo-disk facility available to you, you could use the DOS auto-start facility to invoke this program automatically at cold boot time.

```
A000                * LAST UPDATED 8 NOV 81
A000                *
A000                * THIS PROGRAM SIMULATES A DISK DRIVE IN RAM, IN
A000                * ORDER TO PROVIDE VERY FAST ACCESS FOR SMALL
A000                * FILES.
A000                *
A000                * IT DOES THIS BY INTERCEPTING THE DOS DCOM VECTOR;
A000                * REFERENCES TO DRIVES 1-3 ARE PASSED TO DOS TO
A000                * HANDLE; REFERENCES TO DRIVE 4 ARE HANDLED HERE.
A000                *
A000                * ONLY 1 DIRECTORY SECTOR IS SIMULATED; THE OTHER
A000                * 3 ARE IGNORED.
A000                *
0100                DOSBASE  EQU     0100H       BASE ADDRESS OF DOS
0122                DCOM     EQU     DOSBASE+22H I/O ROUTINE
0128                DOS      EQU     DOSBASE+28H RESTART POINT OF DOS
A000                *
E000                BOOT     EQU     0E000H      MY BOOT PROM
E03C                CRLF     EQU     BOOT+3CH    DISPLAY CR/LF
E066                MSG      EQU     BOOT+66H    DISPLAY MESSAGE AT HL
E069                MSGCR    EQU     BOOT+69H    DISP MSG AT HL, DO CR/LF
E08D                HEXOUT   EQU     BOOT+8DH    DISPLAY ACCUM IN HEX
A000                *
A000                FASTDISK START
A000  21FFBF                 LXI     H,0BFFFH    TOP OF MEMORY
A003  224BA0                 SHLD    MEMTOP      SAVE OLD LIMIT
A006                *
A006  2100A2                 LXI     H,DIREC
A009  CD4FA1                 CALL    FASTENTY    INIT DIRECTORY
A00C                *
A00C                * ALTER DCOM VECTOR
A00C                *
A00C  2A2301                 LHLD    DCOM+1
A00F  2249A0                 SHLD    OLDDCOM+1   SAVE ORIGINAL DCOM VECTOR
A012  214EA0                 LXI     H,FASTRAM
A015  222301                 SHLD    DCOM+1      SET UP OUR VECTOR
A018                *
A018                * DETERMINE SIZE OF RAM BUFFER
A018                *
A018  2A4BA0                 LHLD    MEMTOP      OLD MEMORY LIMIT
A01B  23                     INX     H
A01C  1100A2                 LXI     D,DIREC     START OF RAM BUFFER
A01F  CD67A1                 CALL    SUB16       HL=SIZE IN BYTES
A022  7C                     MOV     A,H         GET NO OF BLKS
A023                *
A023  F5                     PUSH    PSW         SAVE SIZE
A024  219BA1                 LXI     H,MSG2      "SIZE="
A027  CD66E0                 CALL    MSG
A02A  F1                     POP     PSW
A02B  F5                     PUSH    PSW
A02C  CD8DE0                 CALL    HEXOUT      DISPLAY SIZE
A02F  21B2A1                 LXI     H,MSG2B     "H"
A032  CD69E0                 CALL    MSGCR
A035  F1                     POP     PSW         RESTORE SIZE
A036                *
A036  3D                     DCR     A
A037  B7                     ORA     A           CLEAR CARRY
A038  1F                     RAR     .           GET NO OF SECTORS
A039  3C                     INR     A           ALLOW FOR MISSING DIR SECT
                                                 ORS
A03A  3C                     INR     A
A03B  3C                     INR     A
A03C  324DA0                 STA     MAXADDR     SAVE MAX DISK ADDR
A03F                *
A03F  C32801                 JMP     DOS         RETURN TO DOS
A042                *
A042  564543+                ASC     'VECTS'
A047  F1            OLDDCOMV POP     PSW         RESTORE ACCUM
A048  C30000        OLDDCOM  JMP     0           ALTERED TO POINT TO DCOM
A04B  0000          MEMTOP   DW      0           OLD TOP OF MEMORY
A04D  00            MAXADDR  DB      0           NO OF SECTORS IN RAM BUFF
A04E                *
A04E                * ENTERED WHEN DCOM CALLED:
A04E                * B=ACTION CODE: -n = INITIALIZE DISK
A04E                *                 0 = WRITE SECTOR
```

```
A04E              0077 *                     1 = READ SECTOR
A04E              0078 *                     2 = VERIFY SECTOR
A04E              0079 * C=DRIVE + DENSITY; HL=DISK ADDRESS; DE=RAM ADDR;
A04E              0080 * A=NO OF SECTORS.
A04E              0081 *
A04E   F5         0082 FASTRAM  PUSH     PSW        SAVE NO OF SECTORS
A04F   79         0083          MOV      A,C
A050   E67F       0084          ANI      7FH        DROP DENSITY
A052   FE04       0085          CPI      4          ONLY HANDLE DRIVE 4
A054   C247A0     0086          JNZ      OLDDCOMV
A057              0087 *
A057   78         0088          MOV      A,B        GET ACTION CODE
A058   B7         0089          ORA      A          INITIALIZE?
A059   F265A0     0090          JP       FASTCHEK   NO, CHECK STARTING ADDR
A05C   CDB0A0     0091          CALL     CHEKADDR   CHECK STARTING ADDR
A05F   DA9EA0     0092          JC       FASTEXIT   IF PAST END, IGNORE REST
A062   C36BA0     0093          JMP      FASTBYP
A065              0094 *
A065   CDB0A0     0095 FASTCHEK CALL     CHEKADDR   CHECK STARTING DISK ADDR
A068   DAA1A0     0096          JC       ERROR2
A06B   7D         0097 FASTBYP  MOV      A,L        GET DISK ADDR
A06C   B7         0098          ORA      A          WANT FIRST SECTOR OF DIREC
                                                    TORY?
A06D   CA88A0     0099          JZ       FASTFRST   YES, JMP
A070   FE04       0100          CPI      4          WANT REST OF DIRECTORY?
A072   DA2EA1     0101          JC       FASTDIR    YES, JMP
A075   3D         0102          DCR      A          ALLOW FOR MISSING SECTORS
A076   3D         0103          DCR      A
A077   3D         0104          DCR      A          A=DISK OFFSET
A078   D5         0105          PUSH     D          SAVE RAM ADDR
A079   110002     0106          LXI      D,512      SECTOR LENGTH
A07C   2100A2     0107          LXI      H,DIREC    POINT TO RAM BUFFER
A07F   19         0108 FASTMULT DAD      D          MULT BY DISK OFFSET
A080   3D         0109          DCR      A
A081   C27FA0     0110          JNZ      FASTMULT
A084   D1         0111          POP      D          RESTORE RAM ADDR
A085              0112 * HL NOW POINTS TO SECTOR IN RAM BUFFER
A085   C38BA0     0113          JMP      FASTACT
A088              0114 *
A088   2100A2     0115 FASTFRST LXI      H,DIREC
A08B   78         0116 FASTACT  MOV      A,B        A=ACTION TO PERFORM
A08C   B7         0117          ORA      A
A08D   FAC6A0     0118          JM       FASTINIT   -n = INIT DISK
A090   CAD6A0     0119          JZ       FASTWRT    0 = WRITE
A093   FE02       0120          CPI      2
A095   DAF6A0     0121          JC       FASTREAD   1 = READ
A098   CA11A1     0122          JZ       FASTVER    2 = VERIFY
A09B   C3ADA0     0123          JMP      ERROR1     BAD ACTION
A09E              0125 *
A09E   F1         0126 FASTEXIT POP      PSW
A09F   AF         0127          XRA      A
A0A0   C9         0128          RET
A0A1              0129 *
A0A1   CD3CE0     0130 ERROR2   CALL     CRLF
A0A4   2179A1     0131          LXI      H,MSG1     "BEYOND END"
A0A7   CD69E0     0132          CALL     MSGCR
A0AA   C32801     0133          JMP      DOS
A0AD              0134 *
A0AD   F1         0135 ERROR1   POP      PSW
A0AE   37         0136 ERROR    STC      .          INDIC ERROR OCCURRED
A0AF   C9         0137          RET
A0B0              0138 *
A0B0              0139 * VALIDATE DISK ADDRESS IN HL
A0B0              0140 * CARRY SET IF BAD.
A0B0              0141 *
A0B0   7C         0142 CHEKADDR MOV      A,H
A0B1   B7         0143          ORA      A
A0B2   C2AEA0     0144          JNZ      ERROR
A0B5   3A4DA0     0145          LDA      MAXADDR
A0B8   BD         0146          CMP      L          MUST BE <= MAX ADDRESS
A0B9   C9         0147          RET
A0BA              0148 *
A0BA              0149 * ENSURE RAM BUFFER PNTR (HL) STAYS IN BOUNDS
A0BA              0150 * CARRY SET IF BAD
A0BA              0151 *
A0BA   D5         0152 CHEKBNDS PUSH     D
A0BB   EB         0153          XCHG
A0BC   2A4BA0     0154          LHLD     MEMTOP     REAL MEM TOP
A0BF   EB         0155          XCHG     .          HL-->RAM BUFF, DE-->MEM TO
                                                    P
A0C0   CD72A1     0156          CALL     CMPAD
A0C3   D1         0157          POP      D
A0C4   C9         0158          RET
A0C5              0160 *
A0C5              0161 * INITIALIZE DISK
A0C5              0162 * HL-->SECTOR IN RAM BUFFER
A0C5              0163 * DE-->BLANK BUFFER; STACK=NO OF SECTORS.
A0C5              0164 *
A0C5   F5         0165          PUSH     PSW
A0C6   D5         0166 FASTINIT PUSH     D          SAVE SECT ADDR
A0C7   CDE3A0     0167          CALL     FASTWSEC   WRITE A SECTOR
A0CA   D1         0168          POP      D          RESTORE SECT ADDR
A0CB   DA9EA0     0169          JC       FASTEXIT   JMP IF DONE
A0CE   F1         0170          POP      PSW        RESTORE SECT COUNT
A0CF   3D         0171          DCR      A
A0D0   C2C5A0     0172          JNZ      FASTINIT-1
A0D3   AF         0173          XRA      A          INDIC ALL OK
A0D4   C9         0174          RET      .          EXIT TO USER
A0D5              0175 *
A0D5              0176 * WRITE DISK SECTOR
A0D5              0177 * HL-->SECTOR IN RAM BUFFER; A=NO OF SECTORS;
A0D5              0178 * DE-->USER RAM.
A0D5              0179 *
A0D5   F5         0180          PUSH     PSW
A0D6   CDE3A0     0181 FASTWRT  CALL     FASTWSEC   WRITE A SECTOR
A0D9   DAA1A0     0182          JC       ERROR2     JMP IF ERR
A0DC   F1         0183          POP      PSW        RESTORE SECTOR COUNT
A0DD   3D         0184          DCR      A
A0DE   C2D5A0     0185          JNZ      FASTWRT-1
A0E1   AF         0186          XRA      A          CLEAR CARRY
A0E2   C9         0187          RET      .          EXIT TO USER
A0E3              0188 *
A0E3   010002     0189 FASTWSEC LXI      B,512      SECTOR LEN
A0E6   CDBAA0     0190          CALL     CHEKBNDS
A0E9   D8         0191          RC
A0EA   1A         0192 FASTWRLP LDAX     D          GET BYTE FROM USER RAM
A0EB   77         0193          MOV      M,A        WRITE IN RAM BUFFER
A0EC   23         0194          INX      H
A0ED   13         0195          INX      D
A0EE   0B         0196          DCX      B          DONE YET?
A0EF   79         0197          MOV      A,C
A0F0   B0         0198          ORA      B
A0F1   C2EAA0     0199          JNZ      FASTWRLP
A0F4   C9         0200          RET
A0F5              0201 *
A0F5              0202 * READ DISK SECTOR
A0F5              0203 * HL-->SECTOR IN RAM BUFFER; A=NO OF SECTORS;
A0F5              0204 * DE-->USER RAM.
A0F5              0205 *
A0F5   F5         0206          PUSH     PSW
A0F6   010002     0207 FASTREAD LXI      B,512      SECTOR LENGTH
A0F9   CDBAA0     0208          CALL     CHEKBNDS
A0FC   DAA1A0     0209          JC       ERROR2
A0FF   7E         0210 FASTRDLP MOV      A,M        GET BYTE FROM RAM BUFFER
A100   12         0211          STAX     D          STORE IN USER RAM
A101   23         0212          INX      H
A102   13         0213          INX      D
A103   0B         0214          DCX      B          DONE YET?
A104   79         0215          MOV      A,C
A105   B0         0216          ORA      B
A106   C2FFA0     0217          JNZ      FASTRDLP
A109   F1         0218          POP      PSW        RESTORE SECT COUNT
A10A   3D         0219          DCR      A
A10B   C2F5A0     0220          JNZ      FASTREAD-1
A10E   AF         0221          XRA      A          CLEAR CARRY
A10F   C9         0222          RET      .          EXIT TO USER
A110              0223 *
A110              0224 * VERIFY DISK SECTOR
A110              0225 * HL-->SECTOR IN RAM BUFFER; A=NO OF SECTORS;
A110              0226 * DE-->USER RAM.
A110              0227 *
A110   F5         0228          PUSH     PSW
```

```
A111  010002  0229 FASTVER  LXI    B,512      SECTOR LENGTH
A114  CDBAA0  0230          CALL   CHEKBNDS
A117  DAA1A0  0231          JC     ERROR2
A11A  1A      0232 FASTVRLP LDAX   D          GET BYTE FROM USER RAM
A11B  BE      0233          CMP    M          COMPARE TO RAM BUFFER
A11C  C2AEA0  0234          JNZ    ERROR      JMP IF VERIFY FAILED
A11F  23      0235          INX    H
A120  13      0236          INX    D
A121  0B      0237          DCX    B          DONE YET?
A122  79      0238          MOV    A,C
A123  B0      0239          ORA    B
A124  C21AA1  0240          JNZ    FASTVRLP
A127  F1      0241          POP    PSW        RESTORE SECTOR COUNT
A128  3D      0242          DCR    A
A129  C210A1  0243          JNZ    FASTVER-1
A12C  AF      0244          XRA    A          INDIC ALL OK
A12D  C9      0245          RET    .          EXIT TO USER
A12E          0246 *
A12E          0247 * WE ARE ACCESSING MISSING PART OF DIRECTORY.
A12E          0248 * IF WRITE, IGNORE. IF READ, RETURN BLANK
A12E          0249 * DIRECTORY SECTOR. IF VERIFY, RETURN MATCH.
A12E          0250 *
A12E  78      0251 FASTDIR  MOV    A,B        A=ACTION TO PERFORM
A12F  B7      0252          ORA    A
A130  FA9EA0  0253          JM     FASTEXIT   -n = INIT
A133  CA9EA0  0254          JZ     FASTEXIT   0 = WRITE
A136  FE02    0255          CPI    2
A138  DA41A1  0256          JC     FASTFAKE   1 = READ
A13B  CA9EA0  0257          JZ     FASTEXIT
A13E  C3ADA0  0258          JMP    ERROR1     BAD ACTION
A141          0259 *
A141  F1      0260 FASTFAKE POP    PSW
A142  EB      0261          XCHG   .          HL-->USER RAM
A143  57      0262          MOV    D,A        D=NO OF SECTORS
A144  D5      0263 FASTFKLP PUSH   D
A145  CD4FA1  0264          CALL   FASTENTY   DO 1 DIR SECTOR
A148  D1      0265          POP    D
A149  15      0266          DCR    D
A14A  C244A1  0267          JNZ    FASTFKLP
A14D  AF      0268          XRA    A
A14E  C9      0269          RET
A14F          0271 *
A14F          0272 * INITIALIZE DIRECTORY SECTOR AT HL
A14F          0273 *
A14F  1620    0274 FASTENTY MVI    D,32       NO OF DIR ENTRIES
A151  010808  0275 FASTENLP LXI    B,0808H    B=8,C=8
A154  3620    0276 FASTBLNK MVI    M,' '      BLANK FILE NAME
A156  23      0277          INX    H
A157  05      0278          DCR    B
A158  C254A1  0279          JNZ    FASTBLNK
A15B          0280 *
A15B  3600    0281 FASTZERO MVI    M,0        ZERO REST OF ENTRY
A15D  23      0282          INX    H
A15E  0D      0283          DCR    C
A15F  C25BA1  0284          JNZ    FASTZERO
A162  15      0285          DCR    D
A163  C251A1  0286          JNZ    FASTENLP
A166  C9      0287          RET
A167          0288 *
A167          0289 * UNSIGNED SUBTRACT DE FROM HL; ANSWER IN HL.
A167          0290 *
A167  D5      0291 SUB16    PUSH   D
A168  7A      0292          MOV    A,D
A169  2F      0293          CMA
A16A  57      0294          MOV    D,A
A16B  7B      0295          MOV    A,E
A16C  2F      0296          CMA
A16D  5F      0297          MOV    E,A
A16E  13      0298          INX    D
A16F  19      0299          DAD    D
A170  D1      0300          POP    D
A171  C9      0301          RET
A172          0302 *
A172          0303 * COMPARE HL AND DE
A172          0304 * CARRY SET IF HL > DE .
A172          0305 *
A172  7A      0306 CMPAD    MOV    A,D
A173  BC      0307          CMP    H
A174  D8      0308          RC
A175  C0      0309          RNZ
A176  7B      0310          MOV    A,E
A177  BD      0311          CMP    L
A178  C9      0312          RET
A179          0313 *
A179  07      0314 MSG1     DB     7          BEEP
A17A  414343+ 0315          ASCC   'ACCESS BEYOND END OF PSEUDO DISK'
A19B  53495A+ 0316 MSG2     ASCC   'SIZE OF PSEUDO DISK = '
A1B2  482042+ 0317 MSG2B    ASCC   'H BLOCKS'
A1BB          0318 *
A200          0319          ORG    FASTDISK+200H ALIGN TO SECT BOUNDARY
A200          0320 DIREC    DS     512        DIRECTORY AREA
A400          0321 * RAM BUFFER FOLLOWS DIRECTORY
A400          0322          END    FASTDISK
```

SYMBOL TABLE

```
NAME     VALU TY  LEN  REFERENCES

BC       0000  C 0000
BOOT     E000  C 0000   19   20   21   22   23
CHEKADDR A0B0  I 0001  142   91   95
CHEKBNDS A0BA  I 0001  152  190  208  230
CMPAD    A172  I 0001  306  156
CRLF     E03C  C 0000   20  130
DCOM     0122  C 0000   16   34   37
DE       0002  C 0000
DIREC    A200  S 0200  320   29   43  107  115
DOS      0128  C 0000   17   65  133
DOSBASE  0100  C 0000   15   16   17
ERROR    A0AE  I 0001  136  144  234
ERROR1   A0AD  I 0001  135  123  258
ERROR2   A0A1  I 0003  130   96  182  209  231
FASTACT  A08B  I 0001  116  113
FASTBLNK A154  I 0002  276  279
FASTBYP  A06B  I 0001   97   93
FASTCHEK A065  I 0003   95   90
FASTDIR  A12E  I 0001  251  101
FASTDISK A000 EA 0000   25  319  322
FASTENLP A151  I 0003  275  286
FASTENTY A14F  I 0002  274   30  264
FASTEXIT A09E  I 0001  126   92  169  253  254  257
FASTFAKE A141  I 0001  260  256
FASTFKLP A144  I 0001  263  267
FASTFRST A088  I 0003  115   99
FASTINIT A0C6  I 0001  166  118  172
FASTMULT A07F  I 0001  108  110
FASTRAM  A04E  I 0001   82   36
FASTRDLP A0FF  I 0001  210  217
FASTREAD A0F6  I 0003  207  121  220
FASTVER  A111  I 0003  229  122  243
FASTVRLP A11A  I 0001  232  240
FASTWRLP A0EA  I 0001  192  199
FASTWRT  A0D6  I 0003  181  119  185
FASTWSEC A0E3  I 0003  189  167  181
FASTZERO A15B  I 0002  281  284
HEXOUT   E08D  C 0000   23   52
HL       0004  C 0000
MAXADDR  A04D  S 0001   71   63  145
MEMTOP   A04B  W 0002   70   27   41  154
MSG      E066  C 0000   21   49
MSG1     A179  S 0001  314  131
MSG2     A19B  S 0017  316   48
MSG2B    A1B2  S 0009  317   53
MSGCR    E069  C 0000   22   54  132
OLDDCOM  A048  I 0003   69   35
OLDDCOMV A047  I 0001   68   86
PSW      0006  C 0000   47   50   51   55   68   82  126  135  165  170  180
                       183  206  218  228  241  260
SP       0006  C 0000
SUB16    A167  I 0001  291   44
```

THE C′ COMPILER:
It takes a smaller byte out of your micro memory.

Languages like Basic, PASCAL, FORTRAN, and COBOL can deplete your storage capacity long before a program's been completed.

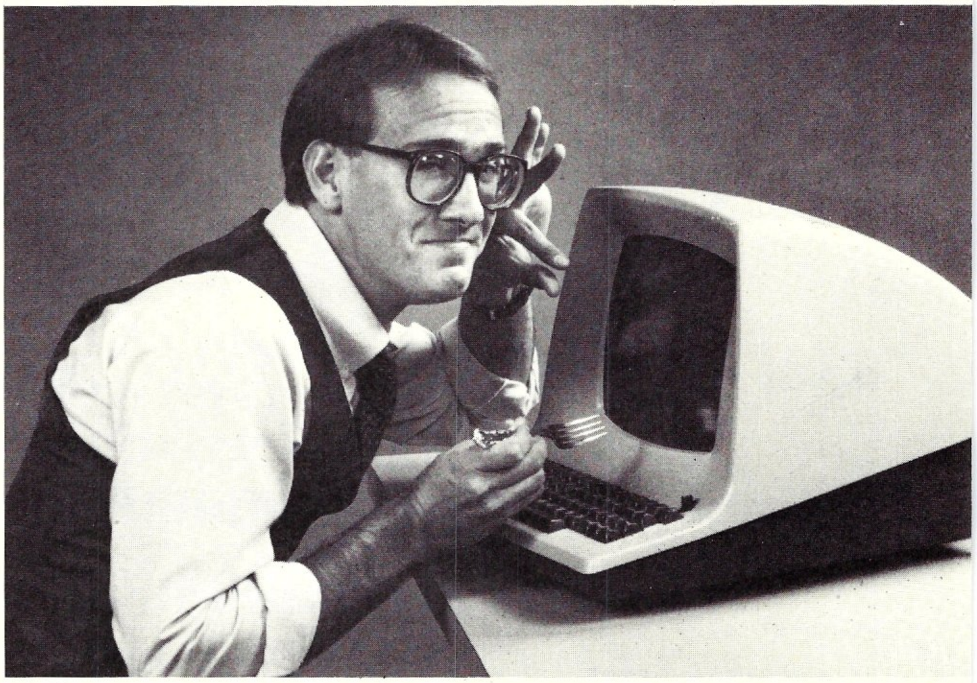
But now Micro Data Base Systems, Inc., has solved the memory problem with the **C′** (C-Prime) Compiler. **C′** generates one of the tightest codes available, so each program uses fewer bytes of your memory. **C′** also offers extremely fast compiler and execution time. And you don't have to give up high-level program and data structuring capabilities. The powerful, general-purpose **C′** language supports structured programming features; data structures; an extensive set of operators; and easy-to-use, assembler-like features.

C′ supports most C language functions, including two-dimensional arrays, structures, recursion, pointers, unions, while, do, for, switch, break, continue, goto, return, labeled statement, signals, and much more. **C′** also comes with a powerful library which implements most of the UNIX Version 7 routines.

A powerful, stand-alone pre-processor executes named file inclusion, macro substitution, and conditional compilation. The compiler itself is written in native code, generates relocatable code, and supports overlaying. The system is compatible with 8080 and Z80 microprocessors, running CP/M and TRSDOS.

The **C′** compiler comes complete with:

- Stand-alone pre-processor
- Disk-to-disk linker
- Standard definition files
- Run-time library
- Assembler-code floating-point library
- Reference manuals
- Sample program
- *The C Programming Language,* by Kernighan and Richie



If you'd like to improve your micro memory, try taking smaller bytes. Send for complete information on the **C′** compiler.

Developed by:

Micro Data Base Systems, one of the world's leading software R&D firms, creators of SCREEN MASTER, **C′**, and other fine software products.

M D B S

Offered by:

International Software Enterprises, an international consortium of the world's leading software and consulting firms, providing developer-friendly tools for building user-friendly applications.

For more information, contact:

In the U.S. and Canada
ISE-USA
85 West Algonquin Road, Suite 400
Arlington Heights, IL 60005
(312) 577-6800

In the United Kingdom and Ireland
ISE-PACTEL 01-828-7744

In West Germany
ISE-ADV/ORGA 089-192-071

In France
ISE-CEGOS 620-61-61

In Sweden
ISE-DATEMA 08 834020

In Norway
ISE-DATEMA 00947 267-0880

In Finland
ISE-TIETOTEHDAŞ 0-5261

In Switzerland and Austria
ISE-ADV/ORGA CH-041-232-360

ELSEWHERE
ISE-INTERNATIONAL (317) 463-4561

ISE-USA, Inc.
85 West Algonquin Road, Suite 400
Arlington Heights, IL 60005

I want to get more from my memory.

☐ Send me more detailed information about the **C′** Compiler.

Name__________ Title__________

Company__________

Address__________

City__________ State__________ Zip__________

Phone (_____) __________ Ext. __________

COMDEX. **See Us At COMDEX—Nov. 29-Dec. 2, 1982 Booth #3260**

CIRCLE 54 ON READER SERVICE CARD

Turnkey +

Fully implemented system-independent warm and cold boot commands for CP/M

by James K. Offenbecher

The ability to turn CP/M into a "TURNKEY" system with automatic program execution on cold or warm boot is a feature that is long overdue for all CP/M operating systems. Several manufacturers have offered this feature on their custom CP/M systems, but in each case the feature has been dedicated to one specific system only. The January 1981 issue of *Microcomputing* (*Kilobaud*) and Zaks' *Handbook of CP/M* both give the locations in the CCP that have to be changed to allow for an automatic warm boot command, but only give a DDT patch, which is tedious to implement. The same is true in Kelly Smith's more recent article in the Jan/Feb 1982 issue of *Microsystems*. Jon Lindsay's version in the January 1981 *Kilobaud* shows a different address for the CCP buffer.

Lindsay's system uses a custom SYSGEN to put the buffer at 0A07H, but standard CP/M SYSGEN version 2.0 puts it at 0987H. Check to see where your SYSGEN puts the buffer if you try to implement this modification. Kelly Smith's article fails to mention the fact that the final JuMP in BIOS to the CCP may have to be modified from 'CCP+3' to prevent clearing the command from the CCP buffer prior to execution.

Why Turnkey?

There are several advantages in having different cold and warm boot commands. First, the system can come up running in an application program. As an example, a CBasic programmer could use different COMMAND variables for the cold and warm boot commands—the cold boot to verify that all necessary files are present, and the warm boot to check the integrity of those files (you should never do a warm boot while chaining programs). But beware! If you use a warm boot command that returns to CP/M with a warm boot, you will not be able to run any other program with that system disk.

Another example would be useful for business programs that put all data on the B drive. By creating a system disk containing only the cold boot command "PIP A:=B:*.*", all data files could be backed up by inserting the backup disk in drive A, pressing the RESET button, and waiting for the system prompt to appear.

Theory

The basic theory behind the modifications is that any command present in the CCP buffer (beginning at location CCP+7) at warm or cold boot times will be executed immediately. The first byte of the buffer (CCP+7) must be nonzero to indicate that a command (beginning at CCP+8) is present, and the command must be terminated by a zero byte. For a command to execute on cold boot, it must exist on the disk but be in the CCP buffer only during the cold boot. The warm boot command must be in the CCP buffer each time a warm boot occurs.

The program and patch described below allow the operator to enter a cold boot command, a warm boot command, or both during a SYSGEN operation when the system is located at a fixed memory location. The patch consists of three parts: (1) An added buffer to hold the cold boot command in the unused portion of BDOS just before the user's BIOS; (2) a small copy routine (run only at cold boot time) in BIOS to replace any warm boot command already present in the CCP buffer with the cold boot command; and (3), the TURNKEY program which inserts or changes the commands in the buffers. The cold boot command is inserted into the added buffer, whereas the warm boot command is inserted directly into the CCP buffer. TURNKEY is run during the SYSGEN operation when the system is in memory at a fixed location. Both commands reside in the system tracks on the disk and can be changed during another SYSGEN/TURNKEY operation. Entering just a carriage return will result in deleting the appropriate command from its buffer. The input routine uses the "Read Console Buffer" BDOS call, which implements all of the CCP editor commands (e.g., ^X,^R).

The fact that BIOS in CP/M version 2.2 allows space for only 380H bytes on a single-density system (Track 1 Sector 26 hold BIOS+300H to BIOS+37FH) adds the additional constraint of space. How many bytes is a cold and warm boot command worth? To alleviate most of this problem, I put the large part (the command buffer) between BDOS and BIOS. Currently, more than 128 bytes are available in CP/M 2.2, due (I assume) to the page mode of addressing BIOS. But by the time BIOS jumps to BDOS during a cold boot, the command is already copied into the CCP buffer and is no longer needed.

Adaptability

This modification should work on any system regardless of density, provided it does not already use a cold boot command to load an extended BIOS, as in some CCS systems.

James K. Offenbecher, 813B Laurel Hill, Fort Dix, NJ 08640

Implementation

The buffer locations that must be used in the TURNKEY program are fixed by SYSGEN. Placing the cold boot buffer outside BIOS renders it independent of any code changes you may make in BIOS. And regardless of memory size configuration, SYSGEN will load the system at the same location in memory. The only common variable is 'BUFLEN', which must be the same in both TURNKEY and the BIOS patch.

The entire BIOS patch with the equates and copy routine should be edited into your BIOS at a point where it will be run only during a cold boot. A good place would be where your BIOS initializes the UARTS, terminal ports or printers. As listed, the patch has 14 bytes of code but may require some register saves as it destroys the PSW, HL and DE registers. If in doubt, save them. The second part of the patch is the actual buffer itself. It is EQUated to be located at a fixed location before BIOS, where it will not be run. The two parameters are optional and may be used after doing a preliminary assembly to insure that the code will fit in the system tracks of the disk. Since the unused space between BDOS and BIOS is zero filled, a modified system run without implementing any commands will only copy a zero to the CCP buffer. Thus no command will be executed.

If you don't have room for the patch in your BIOS, you may attempt to use more of the empty space between BDOS and BIOS, but this can get very tricky to implement and change. Also, much of this space is used for BDOS variable storage once the BDOS is accessed, and will not be available then. With the patches in the BIOS, assemble it and create a new system disk to work with. The modified BIOS system should run normally without implementing the TURNKEY program and should be checked out first.

The TURNKEY program accepts command strings from the keyboard and copies them into the appropriate buffers. Conditional assembly allows the program to be assembled either as an overlay if you use standard SYSGEN version 2.0, or as a stand-alone program if your SYSGEN is different. The code is fairly straightforward and can be assembled using either ASM or MAC. It uses BDOS call 10 to get the command with the character count preceding it, converts lower-case characters to upper case, and then copies it into the appropriate system buffer location using the character count as the nonzero to indicate that a command is present. Set either the ALONE or the SYSGEN equate (but *not both*) to −1. If you changed the length of the buffer in BIOS, the BUFLEN equate in TURNKEY must be changed to match. If you're using the ALONE option, create a .COM file; but if using the SYSGEN option, create only a .HEX file to be linked to SYSGEN version 2.0 with DDT as follows:

```
A>DDT SYSGEN.COM          (Load SYSGEN using DDT)
                          System should respond with:
DDT Vers X.X
NEXT  PC
Ø5ØØ  Ø1ØØ
-SØ32Ø                    (Change two bytes in SYSGEN 2.Ø
Ø32Ø  95  ØØ              so it will link to TURNKEY)
Ø321  01  Ø5
Ø322  CD  .
-ITURNKEY.HEX             (Set up the FCB for TURNKEY)
-R                        (Read it into Memory starting at 5ØØH)
NEXT  PC
Ø5XX  Ø1ØØ
-^C                       (EXIT DDT)
A>SAVE 5 TURNKEY.COM      (Save it on disk)
```

You now have a file called TURNKEY.COM that will ask you for both cold and warm boot commands before each 'DESTINATION DISK' request. Entering only a Carriage Return will produce no command and will delete any command already in the applicable buffer.

If you have a custom SYSGEN, you can either try to patch in the link to TURNKEY or use TURNKEY in its ALONE mode. To execute the stand-alone version, get the system from the disk using SYSGEN, but skip the 'PUT' section. Now run TURNKEY to insert the commands. Finally, run SYSGEN again, this time skipping the 'GET SYSTEM' section.

Testing and problems

With all sections inserted or assembled, try to create a system disk that will cold boot with a STAT and warm boot with a DIR. (Be sure to have STAT on the disk.) If the commands do not execute, get the system from the disk using the original SYSGEN, save it as a file and examine it for the commands with DDT (cold should be just before 1F80H and warm should start at 0987H). If the commands are not in the correct locations or are not terminated by a zero byte, check your added code for errors. If all appears okay but the system just does not see the command, there is one last thing to check. When BIOS completes its system load at cold or warm boot time and jumps to the CCP, if it jumps to CCP+3 it appears to clear the buffer first and thus erases the command there. I had this problem while attempting to implement this mod on a CCS 2422 double-density BIOS and corrected it by changing the JuMP to CCP+3 to be a JuMP to CCP (after quite a bit of head scratching). I am still working on a linking patch for the California Computer Systems CCSYSGEN program.

Miscellaneous

While working on the patch to standard SYSGEN 2.0, I discovered two facts about it that Digital Research failed to document.

The first is a routine in SYSGEN that automatically loads a previously saved version of CP/M from disk into the image area. Enter 'SYSGEN CPM48.COM'. SYSGEN will bring the file 'CPM48.COM' into memory at the proper loca-

The ability to turn CP/M into a "turnkey" system with automatic program execution on a cold/warm boot is long overdue.

Turnkey + continued . . .

tion, skipping the 'GET SYSTEM' prompts. A 'FILE INCOMPLETE' error will occur if the file is too short—but watch out, because a file created with a 'SAVE 8' (or anything above 7) instead of a 'SAVE 34' will not produce this error.

The second may be of help if SYSGEN seems to be very slow in getting and putting the system on the disk. Beginning at Location 012AH is a sector translate table that is factory set to read sectors sequentially (1,2,3,4). For most disk systems this is probably the slowest order to read the sectors. If this is changed to your system's optimum Skew factor it will drastically cut down the SYSGEN Get/Put times. (Mine took about 6 secs with the original and takes less than 1 with a skew of 3). Another word of caution though, the first byte (Loc 012AH) must remain a one or the program will really bomb, so always start with the first sector. Various skew factors are listed below, so start with 6 and go down until a large increase instead of a decrease in times is observed. Use DDT to patch them in, and exit with a SAVE 4 (SAVE 5 if TURNKEY is added).

```
SKEW6 — 1,7,13,19,25,5,11,17,23,3,9,15,21,2,8,14,20,26,6,12,18,24,4,10,16,22
SKEW5 — 1,6,11,16,21,26,5,10,15,20,25,4,9,14,19,24,3,8,13,18,23,2,7,12,17,22
SKEW4 — 1,5,9,13;17,21,25,3,7,11,15,19,23,2,6,10,14,18,22,26,4,8,12,16,20,24
SKEW3 — 1,4,7,10,13,16,19,22,25,2,5,8,11,14,17,20,23,26,3,6,9,12,15,18,21,24
SKEW2 — 1,3,5,7,9,11,13,15,17,19,21,23,25,2,4,6,8,10,12,14,16,18,20,22,24,26
```

```
                ;       T U R N K E Y . A S M

                ; THIS PROGRAM USED EITHER IN CONJUNCTION WITH
                ; ANY CPM SYSGEN PROGRAM OR PATCHED INTO SYSGEN VERS 2.0

                ; IT ALLOWS THE OPERATOR INPUT OF EITHER OR BOTH A COLD
                ; BOOT COMMAND PROGRAM AND/OR A WARM BOOT COMMAND PROGRAM
                ; WHICH CAN BE THE SAME OR DIFFERENT.  IT FUNCTIONS BY
                ; RESERVING A BUFFER SPACE JUST BEFORE BIOS FOR THE COLD BOOT
                ; COMMAND, WITH A SMALL ROUTINE RUN AT COLD BOOT TIME
                ; TO COPY THE BUFFER TO THE CCP BUFFER FOR EXECUTION AT
                ; COLD BOOT.  BY ALLOWING THIS SPACE THIS PROGRAM TRANSFERRS
                ; OPERATOR INPUT TO THIS BUFFER AND DIRECTLY TO THE
                ; CCP BUFFER DURING A SYSGEN OPERATION.  THUS WHEN THE
                ; SYSTEM IS PUT WITH SYSGEN IT CONTAINS THE NEW COLD
                ; AND OR WARM BOOT COMMANDS.  THE OPERATOR NEED ONLY
                ; RUN SYSGEN 2.0 OR A COMBINATION IF NOT 2.0 TO CHANGE
                ; THE COLD AND/OR WARM BOOT COMMANDS.

                ; WRITTEN BY:

                ;       JAMES K. OFFENBECHER
                ;               813 B. LAUREL HILL
                ;                       FORT DIX, NEW JERSEY
                ;                               08640
                ;       TELEPHONE  (609)-723-7529

                ;CONDITIONAL ASSEMBLY EQUATES - SET THE
                ;APPROPRIATE ONE TO BE A -1

0000 =          ALONE   EQU     0       ;SET TO -1 IF STANDALONE
FFFF =          SYSGEN  EQU     -1      ;SET TO -1 IF PATCH TO SYSGEN


                ;       BDOS EQUATES

0005 =          BDOS    EQU     5       ;BDOS CALL LOCATION
0001 =          CONIN   EQU     1       ;CONSOLE INPUT FUNCTION
0002 =          CONOUT  EQU     2       ;CONSOLE OUTPUT FUNCTION
000A =          INLINE  EQU     10      ;READ CONSOLE BUFFER FUNCTION
0000 =          WBOOT   EQU     0       ;WARM BOOT ADDRESS

                ; MISCELLANEOUS EQUATES

000D =          CR      EQU     0DH     ;ASCII CARRIAGE RETURN
000A =          LF      EQU     0AH     ;ASCII LINE FEED
0020 =          BUFLEN  EQU     32      ;LENGTH OF INPUT BUFFER
                ; NOTE:  THE BUFLEN EQUATE SHOULD BE THE SAME AS IN
                ;        THE BIOS PATCH.

                ; MEMORY IMAGE PATCH POINTS
0980 =          CCP     EQU     0980H   ;START OF CCP IN MEMORY
0987 =          CCPBUF  EQU     CCP+7   ;COMMAND LINE IN BDOS
1F80 =          BIOS    EQU     CCP+1600H       ;START OF BIOS
1F5D =          BIOBUF  EQU     BIOS-BUFLEN-3   ;SET UP BUFFER BEFORE BIOS

                ;CONDITIONAL EQUATES (IF PATCHING SYSGEN 2.0)

                        IF      SYSGEN

019A =          MSGOUT  EQU     019AH   ;SUBR LOC IN SYSGEN

                        ENDIF

                ; THIS IS THE STARTING POINT OF THE STANDALONE VERSION
                ; IT SETS THE STACK POINTER UP WHICH IS NOT NEEDED
                ; WITH THE PATCHED VERSION SINCE SYSGEN 2.0'S STACK
                ; IS BIG ENOUGH.

                        IF      ALONE

                        ORG     100H            ;IF STANDALONE ONLY

                        LXI     SP,STACK        ;SET UP STACK POINTER

                        ENDIF

                ; THIS IS THE STARTING POINT IF A PATCH TO SYSGEN IS USED.
                ; SYSGEN'S STACK IS USED AND IT IS ORGED PAST SYSGEN WITH
                ; A FEW BYTES TO SPARE.  TO PATCH SYSGEN  LOCATIONS MUST
                ; BE CHANGED AS FOLLOWS:
                ;       LOC     WAS     CHANGE TO
                ;       0320H   95H     00H
                ;       0321H   01H     05H
                ; USE DDT TO OVERLAY BOOTER.HEX OVER SYSGEN AFTER CHANGES
                ; AND SAVE WITH 'SAVE 5 SYSBOOT.COM'

                        IF      SYSGEN

0500                    ORG     500H

0500 E5                 PUSH    H               ;SAVE H/L FROM SYSGEN

                        ENDIF

                ; COMMON SEGMENT OF PROGRAM STARTS HERE

0501 CD5405     CBOT:   CALL    ZLOOP           ;CLEAR THE BUFFER
0504 218505             LXI     H,CBOOTM        ;POINT TO COLD BOOT REQUEST
0507 CD9A01             CALL    MSGOUT          ;PUT IT ON CONSOLE
050A 11E405             LXI     D,IBUF          ;POINT TO INPUT BUFFER
050D 0E0A               MVI     C,INLINE        ;GET FUNCTION IN C
050F CD0500             CALL    BDOS            ;PUT INPUT IN BUFFER
0512 3AE505     CO      LDA     IBUF+1          ;GET # OF CHARS ENTERED
0515 FE1F               CPI     BUFLEN-1        ;GET # OF CHARS ALLOWED
0517 DA2305             JC      COK             ;BYPASS ERROR IF OK
051A 21BD05             LXI     H,TOLONG        ;POINT TO ERROR MESSAGE
051D CD9A01             CALL    MSGOUT          ;PRINT IT
0520 C30105             JMP     CBOT            ;TRY AGAIN
0523 215D1F     COK:    LXI     H,BIOBUF        ;POINT TO BIOS BUFFER
0526 CD6105             CALL    COPY            ;GO PUT IT INTO THE BIOS
0529 CD5405     WBOT:   CALL    ZLOOP           ;RE-CLEAR THE BUFFER
052C 21A105             LXI     H,WBOOTM        ;POINT TO WARM BOOT REQUEST
052F CD9A01             CALL    MSGOUT          ;PUT IT ON CONSOLE
0532 11E405             LXI     D,IBUF          ;POINT TO INPUT BUFFER
0535 0E0A               MVI     C,INLINE        ;GET FUNCTION IN C
0537 CD0500             CALL    BDOS            ;GO PUT MESSAGE IN BUFFER
053A 3AE505             LDA     IBUF+1          ;GET # OF CHARS ENTERED
053D FE1F               CPI     BUFLEN-1        ;GET # OF CHARS ALLOWED
053F DA4B05             JC      WOK             ;BYPASS ERROR IF OK
0542 21BD05             LXI     H,TOLONG        ;POINT TO ERROR MESSAGE
0545 CD9A01             CALL    MSGOUT          ;PRINT IT
0548 C32905             JMP     WBOT            ;TRY AGAIN
054B 218709     WOK:    LXI     H,CCPBUF        ;POINT TO BUFFER IN CCP
054E CD6105             CALL    COPY            ;GO PUT MESSAGE IN CCP
0551 C37B05             JMP     EXIT            ;EXIT TO CPM OR SYSGEN

                ; THIS ROUTINE CLEARS THE INPUT BUFFER TO ALL ZEROS

0554 AF         ZLOOP:  XRA     A               ;CLEAR A
0555 0E20               MVI     C,BUFLEN        ;SET THE COUNT
0557 21E505             LXI     H,IBUF+1        ;POINT PAST THE MAX LENGTH
055A 77         ZLOOP1: MOV     M,A             ;CLEAR THE BYTE
055B 23                 INX     H               ;POINT TO NEXT
055C 0D                 DCR     C               ;ONE LESS TO DO
055D C25A05             JNZ     ZLOOP1          ;LOOP TILL DONE
0560 C9                 RET

                ; THIS ROUTINE COPIES THE INPUT BUFFER TO THE SPECIFIED
                ; SYSTEM BUFFER CLEARING ANY OLD COMMANDS PRESENT.

0561 11E505     COPY:   LXI     D,IBUF+1        ;POINT TO # OF CHARS INPUT
0564 0E20               MVI     C,BUFLEN        ;PUT COUNT IN C
```

```
Ø566 1A          COPY1:  LDAX    D              ;GET BYTE FROM INPUT BUFFER
Ø567 FE61                CPI     'A'+2ØH        ;LESS THAN LOWER CASE A
Ø569 DA73Ø5              JC      CHROK          ;DON'T NEED CONVERT
Ø56C FE7B                CPI     'Z'+2ØH+1      ;HIGHER THAN LOWER CASE Z
Ø56E D273Ø5              JNC     CHROK          ;DON'T NEED CONVERT
Ø571 E65F                ANI     Ø5FH           ;CONVERT TO LOWER CASE
Ø573 77          CHROK:  MOV     M,A            ;PUT IN OUTPUT
Ø574 23                  INX     H              ;BUMP OUTPUT
Ø575 13                  INX     D              ;BUMP INPUT TOO
Ø576 ØD                  DCR     C              ;ONE LESS TO DO
Ø577 C266Ø5              JNZ     COPY1          ;LOOP TILL DONE
Ø57A C9                  RET

                         IF      ALONE

                 ; THIS ROUTINE OUTPUTS A MESSAGE TO THE CONSOLE DEVICE
                 ; BY CHARACTER UNTIL A ZERO IS ENCOUNTERED.
                 ; ALSO INCLUDES STANDALONE EXIT ROUTINE.

                 MSGOUT: MOV     A,M            ;GET BYTE IN A
                         ORA     A              ;SET Z FLAG IF END
                         RZ                     ;RETURN IF DONE
                         PUSH    H              ;SAVE THE POINTER
                         MOV     E,A            ;PUT CHAR IN E FOR BDOS
                         MVI     C,CONOUT       ;GET FUNCTION IN C
                         CALL    BDOS           ;GO PUT THE CHAR ON CONSOLE
                         POP     H              ;RESTORE POINTER
                         INX     H              ;POINT TO NEXT BYTE
                         JMP     MSGOUT         ;LOOP TILL DONE

                 EXIT:   JMP     WBOOT

                         ENDIF

                         IF      SYSGEN

Ø57B 21E1Ø5      EXIT:   LXI     H,NEWLIN       ;POINT TO NEW LINE MESSAGE
Ø57E CD9AØ1              CALL    MSGOUT         ;PRINT IT
Ø581 E1                  POP     H              ;RESTORE H/L FROM SYSGEN
Ø582 C39AØ1              JMP     MSGOUT         ;JUMP BACK INTO SYSGEN

                         ENDIF

                 ; SYSTEM MESSAGES

Ø585 ØDØA454E54CBOOTM:   DB      CR,LF,'ENTER COLD BOOT COMMAND',CR,LF,Ø

Ø5A1 ØDØA454E54WBOOTM:   DB      CR,LF,'ENTER WARM BOOT COMMAND',CR,LF,Ø

Ø5BD ØDØA544F4FTOLONG:   DB      CR,LF,'TOO MANY CHARACTERS - TRY AGAIN',CR,LF,Ø

Ø5E1 ØDØAØØ      NEWLIN: DB      CR,LF,Ø

                 ; INPUT BUFFER STORAGE

Ø5E4 2Ø          IBUF:   DB      BUFLEN  ;BUFFER LENGTH FOR BDOS CALL
Ø5E5                     DS      BUFLEN  ;THE ACTUAL STORAGE LOCS

                 ; STACK SPACE IF STANDALONE ONLY (USE SYSGEN STACK IF OK)

                         IF      ALONE

                         DS      31
                 STACK:  DS      1

                         ENDIF

Ø6Ø5                     END
```

```
;       BIOS PATCH FOR COLD BOOT COMMAND EXECUTION
;
;*************************************************************
;THE FOLLOWING CODE SHOULD BE ADDED IN THE COLD BOOT SECTION
;OF BIOS BEFORE THE JUMP TO THE CCP IS MADE.  IT MAY BE LOCATED
;IN THE DEVICE DRIVER INITIALIZATION SECTION OF YOUR BIOS OR IN
;ANY SECTION OF BIOS THAT IS OVERWRITTEN BY BUFFERS AND CALLED
;FROM THE COLD BOOT SECTION BUT UNDER NO CIRCUMSTANCES SHOULD
;IT BE LOCATED SO THAT ANY PORTION OF IT OCCUPIES SPACE ABOVE
; BIOS + 38ØH AS THIS SPACE MAY EXIST IN MEMORY BUT IT DOES
;NOT EXIST ON A SINGLE DENSITY SYSTEM.(TRACK1,SECTOR 26 HOLDS
;MEMORY FROM BIOS+3ØØH TO BIOS+37FH).
;USE THE CODEND AND DSKEND EQUATES TO INSURE THAT IT WILL FIT.
;CHECK THE REST OF THE CODE TO INSURE THAT IT DOESN'T PASS
;DSKEND TOO.
; ALSO CHECK TO SEE IF ANY REGISTERS NEED TO BE SAVED AS
;THIS ROUTINE AFFECTS PSW, D/E, AND H/L.  THEY SHOULD BE
;PUSHED PRIOR AND POPPED AFTER IF IN DOUBT.
; ALSO CHECK THE END OF YOUR BIOS COLD BOOT ROUTINE TO SEE IF
;IT JUMPS TO CCP OR TO CCP+3.  IF IT GOES TO CCP+3 IT WILL
;CLEAR THE COMMAND FROM THE BUFFER BEFORE EXECUTING IT, AND
;MUST BE CHANGED TO JUST 'CCP'.
;
;**********  CODE STARTS HERE  **************
;
; OFFSET USED FOR COLD BOOT MESAGE
CCPBUF  EQU     CCP+7    ;LOC TO PUT COLD BOOT COMMAND
BUFLEN  EQU     32       ;OR WHATEVER YOUR SYSTEM REQUIRES
                         ;SHOULD MATCH THE OTHER SECTIONS
;
; THE ACTUAL COLD BOOT BUFFER IS LOCATED IN THE UNUSED
; SECTION OF BDOS JUST BEFORE BIOS'S ORG LOCATION.
; EVEN IF BDOS NEEDS THIS AREA, IT IS ONLY USED BY BIOS
; BEFORE THE JUMP TO BDOS, AND AT THAT TIME IT MAY BE
; OVERWRITTEN SINCE IT WILL BE RELOADED FROM DISK IF NEEDED.
;
MESSAG  EQU     BIOS-BUFLEN-3    ;PUT MESSAGE IN UNUSED
                                 ;SECTION OF BDOS
;
; COPY STRING ROUTINE FROM BIOS BUFFER TO CCP BUFFER
; ONLY DURING COLD BOOT.
;
        LXI     H,CCPBUF         ;POINT TO CCP BUFFER
        LXI     D,MESSAG         ;POINT TO MESSAGE
MSGLOP: LDAX    D                ;GET BYTE
        MOV     M,A              ;PUT IT
        INX     D                ;NEXT SOURCE
        INX     H                ;NEXT DESTINATION
        ORA     A                ;DID WE DO ZERO
        JNZ     MSGLOP           ;LOOP TILL WE DID
;
CODEND  EQU     $                ;END OF CODE
        ;NOTE:  CODEND MAY BE PUT AT THE END OF YOUR
        ;BIOS CODE TO GIVE YOU AN INDICATION THAT IT
        ; WILL OR WILL NOT FIT ON THE SYSTEM TRACKS.
;
DSKEND  EQU     BIOS+38ØH        ;END OF SYSTEM ON DISK
        ; IF THE VALUE OF 'CODEND' IS GREATER THAN
        ; THE VALUE OF 'DSKEND', THEN THE CODE
        ; WILL NOT FIT ON A SINGLE DENSITY SYSTEM DISK.
;
;**********************************************************
; END OF MODIFICATIONS
```

HARDWARE REVIEW

Twenty-six Megabytes for Your Computer

A hardware review of the Morrow Designs M26 hard disk drive, controller, and software package

by Paul H. Earley

How often have you asked yourself, "What if I had a hard disk?" I am willing to bet that most S-100 computer owners, from multiuser business system operators to "hackers" with a hand-soldered Altair computer, have thought about installing a hard disk at one time or another. The reason that fantasy has not become a reality for most of us is cost: A typical S-100 complete hard disk system runs from $2,700 to $12,000. Winchester back-up is not included at such a price. Prices *are* falling, however, and with the hardware competition continuing at its current rate, soon all micros might contain a hard disk. The second dream-shattering reality most S-100 owners have concerns the question of bus compatibility. The phrase "meets or exceeds all IEEE-696 standards" is often met with scepticism. And, if you are like most users, at least half of your cards suffered regulator burn before IEEE-696 was a gleam in the eyes of Messrs. Elmquist, Fullmer, Gustavson, and Morrow. Who is to say that your boards will function with this fancy protocol anyway? I should like to address many of these issues in the form of a product review of the Morrow Designs M26 Hard Disk System.

Briefly, at the outset, I should like to inform you of what you will *not* find in this article. First, this is not a "how to build your own Winchester interface" article, on the hardware *or* software level. I will discuss how to implement a complete product into your complete (minus hard disk) computer. Second, although I make no pretense that this is a tutorial on current hard disk technology, I will discuss some ideas that may help the systems purchaser decide on the type of hard disk that will suit him best. Third, this is not a hype for Morrow Designs, Inc. We paid the full price (and then some) to install our hard disk. And fourth, although the article will spend a bit of time discussing the intricacies of the software interface, I cannot address all possible system configurations. If you or an associate are not adept at 8080/Z80 assembly language programming, then the customization of the Morrow Designs hard disk is not for you. It helps to have brought up at least one floppy-based computer, as the subtleties of high speed data transfer baffle virtually every programmer. (I should note that, as of this writing, the Morrow Designs disk system software is set to "drop in" to a computer that uses 98% Morrow products).

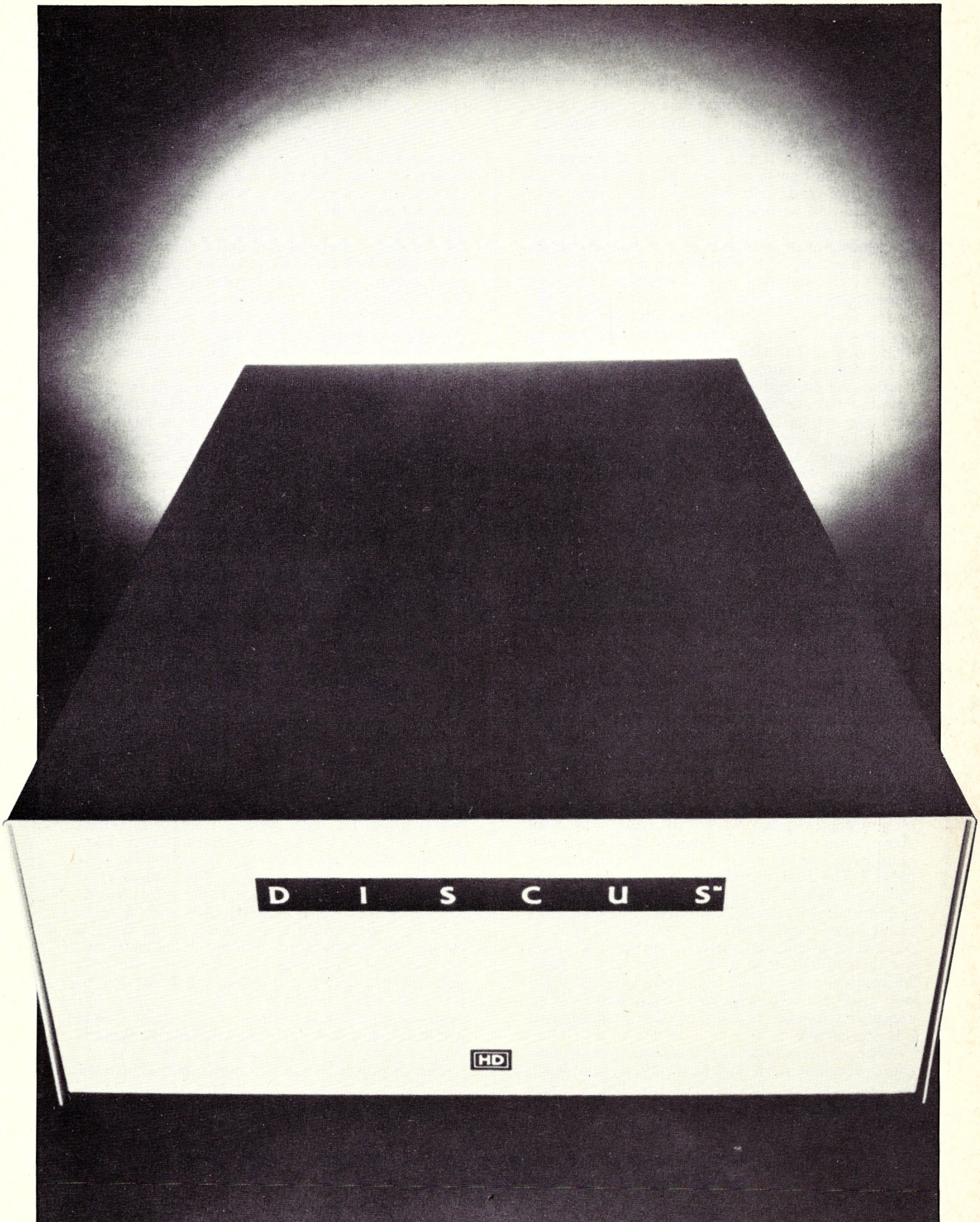
Paul H. Earley, M.D., 2904 S.E. 28th Ave., Portland, OR 97202

Without stepping past my bounds, I feel it would help the potential M26 (or M20, M10) customer to develop a flavor for the Discus 26 Megabyte Hard Disk System (hereafter called the M26). Note that the M10 and the M20 use the same controller and, with the exceptions noted below, can be thought of as almost identical systems. The M26 has no packaging frills, does not have brushed chrome name plates on the drive chassis, and the controller board comes populated with only the chips necessary for the number of drives you buy. Overall, it appears solid in construction. The documentation (covered below) is sufficient, but just so. The hardware design is sound, without "bells and whistles," to run effectively in the S-100 milieu. Attention seems to have been paid more to reliability than to speed (see below). The controller card has little in the way of "jumperable options" that seem so popular these days. The product comes assembled and tested only. The board layout and construction is aesthetically clean and the construction, down to the soldering, is pristine. The card revision we have (HDCA-3) has *no* provision for addressing the extended address lines to transfer data to RAM above 64K. In summary, the M26 is a solid, no-frills hard disk system with somewhat limited upgradability.

Hard vs. floppy disks

With this brief background of one Winchester technology storage system, it seems appropriate to provide the potential hard disk buyer with a few essentials defining what a hard disk is. As the cost of a hard disk subsystem is significant, one spends quite a bit of time shopping before one buys. Everyone who reads *Microsystems* has some understanding of what a hard disk entails. The current boom in production and integration of hard disks in the more inexpensive computers is a result of the so-called "Winchester" technology, introduced in 1971 by IBM with the 3040, followed by a nonremovable media drive, the 3050. Floppy drives had their infancy at much the same time. Floppy disks, due to lower production costs, have enjoyed greater sales.

Floppy and hard disks are very similar in many


DISCUS™
HD

respects. Both have a round disk or platter that is driven from the center. The top and bottom surfaces of the disk are coated with metal oxide magnetic compounds, similar to those used for recording tape. Any given area on that disk can be accessed by a read/write magnetic head. Some floppies read and write data on both sides of the removable diskette. As the cost of the read/write head mechanism is, relatively, the cheapest aspect of the hard disk, all Winchester drives use both sides of the spinning platter for data (although some multiplatter hard drives use one side of one platter for "servo" or alignment information). The first essential difference between the hard and the floppy disk is the speed of disk rotation. The speed of rotation in a floppy is limited by the amount of friction the protective jacket and the head convey onto the disk surface. The rotational speed can be much greater in the hard disk because they have no "jacket" when in place, and the head does not actually touch the disk. An 8″ floppy drive rotates at 360 rpm, whereas a typical hard disk rotates at between 2964 and 3600 rpm.

The method by which fluctuations in magnetic energy are detected constitutes the second major difference between a hard and a floppy drive. A floppy disk functions in essentially the same manner as an audio tape device; that is, the read/write head pushes up against the media. Changes in magnetic flux create small signals that are amplified and decoded into data. Obviously, this constant friction tends to wear down the disk. Efforts to prolong media life (along with other factors) wind up limiting how close together different signal tracks can reside. This density limitation is overcome by manipulating the method by which data is sent and retrieved by the drive controller, but a finite upper limit to data packing soon occurs (this upper limit has not yet been reached in floppy technology, however).

In contrast, Winchester technology floats, or more properly stated, "sails" the read/write head above the surface of the magnetic media. Friction plummets, and as a result the media life is extended. The packing density of data is increased (given as the number of bits that make up the circumference of a track, expressed in FCPI [flux changes per inch], and as the number of concentric circles on a given platter, expressed in tracks per inch [TPI] along a radius line on the disk). Theoretical rate of data transfer is increased and wait time to reach a sector is decreased in Winchester technology, because the disk is spinning at a higher speed.

With all these lovely attributes going for it, how come every home does not have a hard disk? Well, the final contrast between hard and floppy drives turns out to be the most important. Turning a hard platter at 3600 rpm and sailing a R/W head above a rapidly rotating platter requires expensive hardware. Most importantly, ensuring that dust, pollen, cigarette smoke, or even your beard hairs do not get between the head and the media requires that the air surrounding the platters be very clean. This is really the only reason that Winchester technology drives do not include removable media (two manufacturers have released data on removable-media Winchesters, but these have yet to be incorporated into a commercial S-100 system). The rotating platters move in air cleaned by an "absolute filter," and the critically clean parts of a Winchester are assembled in special "clean rooms." This is basically why you get taken to the cleaners when you shell out for a hard disk system.

The third difference between the hard and floppy drive is that, in general, more decoding and signal processing occurs in the bulk of the hard disk unit itself, rather than in the controller card. This relieves the system of the brunt of the data encoding and decoding, a major portion of the electronics design in any type of storage system. The extent of drive intelligence varies; for example, the Century Data Intelligent Marksman expects commands as complex as "format the disk," etc.

Hard disk drives and interfaces

Now we know how hard disks are alike. Some differences between drives can make a difference in whose S-100 system you buy. However, most of the decisions have already been made for you. Morrow Designs ships the M26 with the 14″ Shugart SA4008; the M20 with the Memorex 102 or its Fujitsu second-source equivalent, the 2302; and the M10 with the Memorex 101 (or its Fujitsu equivalent, the M2301B). Their 5-megabyte system uses the popular Seagate Technology ST-506. Different manufacturers have chosen sides with other hard disk manufacturers. Ades and one of the Konan controllers use the high technology Priam drives for their system. CompuPro (Godbout), as of this writing, is distributing their hard disk controller without drives; it is compatible with the Seagate ST-506 and SA1000 (Shugart 8″ drives) type of interface. See Table 1 for a comparison of some of the more popular hard disk systems for the S-100 bus. The basic difference among the smorgasbord of drives is data transfer rate (which is deceiving), the size of the drive itself, and the head driver mechanism. These are the same old "this is better than that" arguments you heard bandied about with the floppy disks. What winds up being the single most important ingredient in most serious users' minds is "Will it last?"

The second and more integral part of a hard disk system is the controller card. Cards vary considerably in their complexity, but the largest single

The hardware design is sound, without "bells and whistles." Attention seems to have been paid more to reliability than to speed.

difference is whether or not the controller uses TMA or I/O mapping to effect communication with the S-100 system (I will use TMA, for Temporary Master Access, in future references to S-100 burst mode I/O-to-memory transfers, a.k.a. DMA). I know of no memory-mapped Winchester controller. Most hobbyist computer owners shy away from TMA systems due to horror stories of incompatibility between dynamic memory devices and TMA controllers. The basic advantage of a TMA I/O device is that a burst of data can be transferred at a rate determined by the peripheral device. This becomes especially important when that device can supply data faster than the CPU software overhead can handle it. In double density with an 8″ floppy disk drive, a 4MHz Z80 CPU is strained to transfer sectors of data. This means that any speed of transfer which is faster requires TMA. In the real world, it is often easier to use an internal transfer buffer (as Morrow Designs does) or accept information at a slower rate, rather than implement a true TMA device for the S-100 bus. Comparisons between the various products are seen in Table 1. I should note that I purposely did not include data transfer speed comparisons; this is a true can of worms. As with all hard numbers, how the data is collected changes the figures. In general, TMA devices are faster. The true sense of speed to the end user, however, is greatly affected by the operating system. My comments about the M26 are given below.

Installing the M26 hardware

We felt that the Morrow Designs M26 was the best investment for our money on the major premise that many of the units were in use in the field, and it was quite economical. We needed a hard disk to allow rapid, real-time acquisition of bioelectric signals for later analysis. Both the speed and the storage capacity of our current computer were insufficient. The S-100 system we have consists of a Cromemco SCC CPU and I/O card, two 32K Delta static memory cards, the Tarbell double density DMA disk controller with two SA800 disk drives, the D+7A analog-to-digital card from Cromemco, and a few wire-wrapped clock/calendar and parallel I/O Cards. I called Morrow, concerned that the M26 would not be compatible with our system. They had no direct experience with a set-up such as ours. So, we ordered the M26 (spending about $4,000) and hoped that no hardware conflicts would occur.

Now for a blow-by-blow description of the effort we have had with the M26. The package was bought through a large mail order house with which we have dealt in the past. Upon the arrival of not one, but three large boxes, we considered our computer space carefully. The size difference between an 8″ and 14″ drive is considerable. The Shugart SA-4008 came in a well-protected box that appeared to be straight from the Shugart factory (then how did all that software get on the hard disk?). The second large box contained the chassis/power supply, and the third box contained all the other goodies, including cables, documentation, the controller card, etc. All three boxes were excellent in construction, built to be used again if needed.

Somewhere in the documentation there are two pages of assembly instructions for the chassis. The assembly of the drive into the chassis is quite simple, but should be done at the installation site of the main computer as the hard disk's protective struts are removed. Once unpacked, the mounting hardware for the SA-4008 is bolted into the chassis with conventional tools (a screwdriver and an 11-mm wrench are all that's needed). The power supply connections and signal connections take a minute to connect and double-check, and the chassis layout is very straightforward and solid in construction. The signal cables run from the drive itself to connectors on the rear of the chassis, permitting cables to be disconnected from the housing without opening the lid—a nice touch.

Now, move the drive to its resting place. The drive belt/platter screw, which prevents aberrant platter rotation during shipping, is removed first (*before* applying AC power). Then, once the SA-4008 is sitting in its permanent location, power is applied, and a clip that prevents wanton head movement is removed (once the disk is turned on). This completes the simple set-up of the hard disk/chassis assembly. Once powered up, the cooling fan impressed me as being quite loud, the one small complaint I have to this day.

The controller card is simple in its installation as well. As I mentioned above, there are few card options. The controller communicates with the CPU via four I/O ports, and the one switch determines the base address of the controller card. The "standard" address is 50 hexadecimal. The boot software on the disk assumes this base address. The only other option on the switch is a "board enable" switch that seems of little use to me. There is provision for the generation of two interrupts, which can be wired in if needed. The interrupts concern two signals OPDONE (signaling completion of a data transfer command) and SDONE (signaling completion of a seek). Provisions are given to tie these two signals to any of the vector interrupt lines, VI0 to VI7, or to INT. Use of these signals in an interrupt environment will

More signal processing takes place in the hard disk unit than in the controller . . . this relieves the system of the tasks of data encoding/decoding.

greatly increase throughput. CP/M has nothing to do with interrupts for data transfer. We intend to use them for data sampling and analysis.

Once the I/O address is set, the cables must be connected to the card. The printed legend and the instruction manual do not help one decide where to plug in the cables. For a one-drive system, the 50-conductor control signal cable and the 20-conductor data cable run directly from the controller board to the chassis of the disk drive. There is only one 50-conductor header on the S-100 card, but it does have positions for four 20-conductor data cables. Physical drive #1 is the header *closest* to the 50-conductor header. This is a small problem, and one can trace the schematic to discover the correct header to use, but, as shipped, the printed legend "P2" on the card lacks clarity, and the manual does nothing to help you decide which 20-pin header to use.

Installing the software

After the electrical connections are made, the larger, more arduous job of software interconnection needs to be done. As shipped, the CP/M BIOS (called CBIOS&ASM) is capable of supporting the Discus floppy and hard disk systems only (with a number of console I/O options). As mentioned above, we have the Tarbell double-density controller, working in a deblocked 512-byte sector mode. This created a double problem. Morrow, in an attempt to ship a lot of software on one disk, sent the source programs on a floppy *unreadable* by our controller. Certainly it is not asking too much to send software in a readable form, and as the Morrow M26 is being shipped to a great number of systems integrators it makes sense to at least ship the needed software on an 8″ single-density disk (or two single-density disks). Very few controllers can access all sector sizes, and some of us still are running in single density.

I called Morrow the day following the receipt of my hardware, and was referred to the customer support people. They did talk with me *curtly* and informed me that it would cost an additional $25. I must say that they treated me fairly, but, as many hardware manufacturers seem to do these days, it sounded as if they felt I was ignorant of most anything that had to do with hardware. It seems a bit silly that this was necessary. While awaiting the software, I hand-entered the short hard disk boot routine. The listing for this simple 8080 assembler routine is listed in the manual. Following its execution, I looked through memory and *voilà!* I found the CP/M logo sitting in memory at the correct location for a 28K CP/M operating system. By patching jumps to my own BIOS at 63K, I was able to implement the hard disk system. In effect, I had two versions of CP/M in memory: The version "SYSGENed" on the hard disk addressed the hard disk as drives A, B, & C; while my own CP/M, sitting higher in memory, had two floppies addressed as A & B.

As the system in such a state was awkward at best, I welcomed the arrival of the CBIOS from Morrow. Once it arrived, it was necessary to strip all assembly not related to the hard disk out of the listing. My goal was to implement the system having drives A and B as my floppies, and C, D and E as the hard disk. (No, I don't have three hard disks. Read on—CP/M 2.2 pushes limitations onto the M26.) This was time-consuming to say the least. My BIOS philosophy was somewhat primitive: I hoped to enter each disk-related basic BIOS function (say, home selected drive) and have a flag set aside in the BIOS that would know which portion of the BIOS to use (the Morrow or the Tarbell) and would jump to the appropriate routine. The flag byte was set by the BIOS function SELDRV (select drive).

This method wound up repeating many functions. The SETSEC (set the next sector to access) routine, for instance, sets one value for my floppy controller, and I simply added code to send that data to the M26 controller as well. Simple. What I wound up producing was a *large* assembly source. It required reducing system size to 61K, so as to make more RAM available above BDOS for the enlarged BIOS. This wasn't too shabby, as the Tarbell deblocked BIOS required an extra 1K, and adding the hard disk I/O took up an additional 2K (which included a second 512-byte sector buffer). All the sector blocking and deblocking routines (necessary, as the Shugart SA4008 is formatted from Morrow for 512-byte sectors) are repeated in my CBIOS, as their implementation by Morrow and Tarbell takes quite different forms. The size of the CBIOS could have been substantially reduced by generating a common deblock routine, but I felt the system development time was not worth the savings of 1K of memory space.

After all this work, it seemed to me that Morrow would do well to include a source file that allows the M26 to "tack on" to any BIOS. Such a generic file was what I had indeed produced. Instead, the CBIOS they sent is really a modification of their CBIOS for floppies (the Disk Jockey controller) with assembly options for the hard disk. Well, once the modifications were made, I produced a workable system. If anyone out there needs the "generic" BIOS as an add-on to their floppy BIOS, I can send it to you.

The main advantages of a hard disk are speed and available disk space. The product is a solid system that performs well . . . . The documentation could use some improvement.

Up and running with the M26

I created a few bugs in the hard disk CP/M CBIOS by modifying it to work with the Tarbell controller. Once these were straightened out, I was up and flying. As to the main advantages to having a hard disk: speed and available disk space—I love them. Eight megabytes is the maximum size of storage device addressable by CP/M. The solution to using physical storage systems of greater capacity entails dividing the physical disk into several "logical" disks. The M26 is separated into three "logical" 8-megabyte drives designated C, D, and E in my system. I tend to visualize it in my head as it is on the disk: The tracks in the outer one-third comprise drive C, second one-third is D, etc. The 8-megabyte limitation is somewhat of a blessing. As it is, to allow sensible directory listings, we make use of the "User" function of CP/M 2.2. For our future data analysis, it seems that any given temporary experiment storage will not exceed 6 megabytes of data; thus CP/M is not preventing us from a simple implementation by this constraint.

The speed of the M26 is an interesting subject. I guess I expected instant data transfer when we installed the hard disk. The Morrow M26 "seems" to be about three to four times as fast as the Tarbell controller under CP/M directed by *my* CBIOS. Executing a program such as MAC, Digital Research's macro assembler, the introduction logo appears on the screen almost as fast as you look up to the screen from the terminal keyboard. This article, about 29Kbytes, is saved with WordStar® in about 2 to 3 seconds. Some data transfers, such as PIP with read after write verification (option 'V'), seem to slow the controller down, but it still "feels" about three to four times as fast as the Tarbell controller.

At present, we are writing an interrupt-driven data storage algorithm for the M26.

Our M26 has been in use for a month now, at about 40 hours a week. We have never seen a data error. Once a sector develops media errors, unlike floppies, you can't throw it out. To save the hard disk, bad sectors are locked out of further use. Morrow has thoughtfully included three programs that set aside any sectors that develop errors. We have yet to use the program for that purpose, but the program has been used to test for the development of bad sectors. The disk contains a single-page file explaining how to use the sector testing and "set aside" programs, which seems sufficient documentation. Included in the many other software bonuses one gets with the M26 is a program called FORMATHD, which formats the hard disk, but will do hard disk diagnostic tests as well. It is self-prompting, but *be careful:* The program does not tell you which of the diagnostics destroy data on the disk, and which do not. Oh, and a copy of Microsoft Basic is included in the purchase price, as is a copy of CP/M 2.0. This bunch of software is helpful, as are several other listings, among them a source for a hard disk CP/M boot routine, if you would like to boot straight up from the hard disk (the SA-4008/M26 requires a two-*minute* warm-up time prior to boot). Also, an 8080 source listing of the very basic hard disk I/O functions is provided to assist the production of Disk I/O through routes other than CP/M.

Table 1. Comparisons between S-100 hard disk systems

Company	Product name	Hard disk used	No. of drives/ Megabytes per drive	I/O type	Suggested list price	Meets IEEE Std.?	Comments
Morrow Designs	M26	Shugart SA4008	4/26	I/O mapped	$4,495	Yes	Many units installed
	M20	Memorex 102 Fujitsu 2302	4/20	I/O mapped	$4,495	Yes	
	M10	Memorex 101 Fujitsu 2301	4/10	I/O mapped	$3,695	Yes	
Konan	SMC-100	Control Data 9448 & others	4/32, 66,96	DMA	$9,800 to $12,500	Yes	Much OEM use; on market 2 years
	DGC-100	Seagate ST-506					

Documentation

You will note that I have said little about the manual for the M26. It has a "preliminary" title on it, spends time discussing the basic disk I/O functions, but really gives little to help implement the M26. It does indicate to the purchaser that software companies and computer clubs often are good sources for help with implementation.

Thanks. It would help to have included in the manual some additional discussion of the ancillary software included in the package. In general I would rate the construction guide as brief but sufficient, and all additional manuals as poor. In direct contradistinction, the source program CBIOS is excellent in its comments, and this alone helped me get the M26 up and running on a nonstandard system.

Conclusion

In summary, the M26 hard disk controller is a solid system that performs well. It does lack somewhat in its ability to grow with your system, as it cannot address more than 64K (obviously, it would work with a bank-select system). The product did *exactly* what it was advertised to do. The documentation, in the current state of microcomputer hardware, could use a bit of improvement. We are very happy with the feel of our system; it is rock solid after one month's use. I only wish we had waited six months, as Morrow now sells a TMA controller for little additional money.

Dr. Paul H. Earley is currently applying microcomputers to the work of real-time collection and analysis of bioelectric signals at the Regional Sleep Disorders Center in Portland, Oregon. His work also includes off-line analysis (by fast Fourier transform) of electroencephalographic data, and the ergonomics of inputting graphic data.

Teleram S-100 Bubble Memory System

A nonvolatile memory for use in hostile environments that make floppy disks unreliable

by Sol Libes

Two and three years ago we heard a lot about how solid-state bubble memory was going to replace floppy and hard disk systems. After all, floppy disk systems are basically electromechanical systems and we all know that mechanical systems are delicate, wear out, have poor reliability, take up a lot of physical space and require a considerable amount of electrical power. Bubble memory systems therefore promised to end our dependency on these electromechanical systems and provide lower-cost, lower-power, more reliable, more rugged, more compact mass memory systems.

Unfortunately the promise of bubble memory has not been fulfilled— or at least not fulfilled completely. The problem is sort of like the chicken-and-the-egg situation. While bubble memory systems have gone into production, floppy and hard disk storage systems have improved dramatically and prices have decreased even more dramatically. The floppy disk market has grown tremendously, causing companies to invest heavily in further disk development and manufacturing capability, product improvement and cost reduction.

The result is that bubble memory has found it difficult to get started in the marketplace. Until there is some meaningful acceptance of the product there will be no significant production and no meaningful reduction in price. And until there is a reduction in price, bubble will not be able to compete with disk in the mass market. Further, until sales revenues increase, manufacturers will be unwilling to invest in further bubble R&D.

What all this means is that bubble will probably never be able to catch up and overtake disk the way the experts predicted. At this time it does not look as if bubble will ever make it as a mass-market data storage system.

But bubble memory systems do have advantages over disk systems. And there are many applications where mass storage is required and disk systems cannot do the job, particularly where disk storage systems would be unreliable. Thus one might consider using a bubble memory system in applications where the disk system would be subjected to heavy dust, very harsh environments (e.g., aboard ship), in portable systems subjected to mechanical shock, and where compact size is important. It is for these applications that the user should take a close look at using a bubble memory system and evaluate whether the high cost is justified by the advantages it offers.

The Teleram S-100-MBMS is the first S-100 magnetic bubble memory card to come on the market. In fact, it presently has no competition. At a list price of $1,945, it is almost twice the cost of a comparable floppy disk system. But then again it is completely solid state, fits into one S-100 card, consumes low power (only 20 watts per card) and is nonvolatile as well.

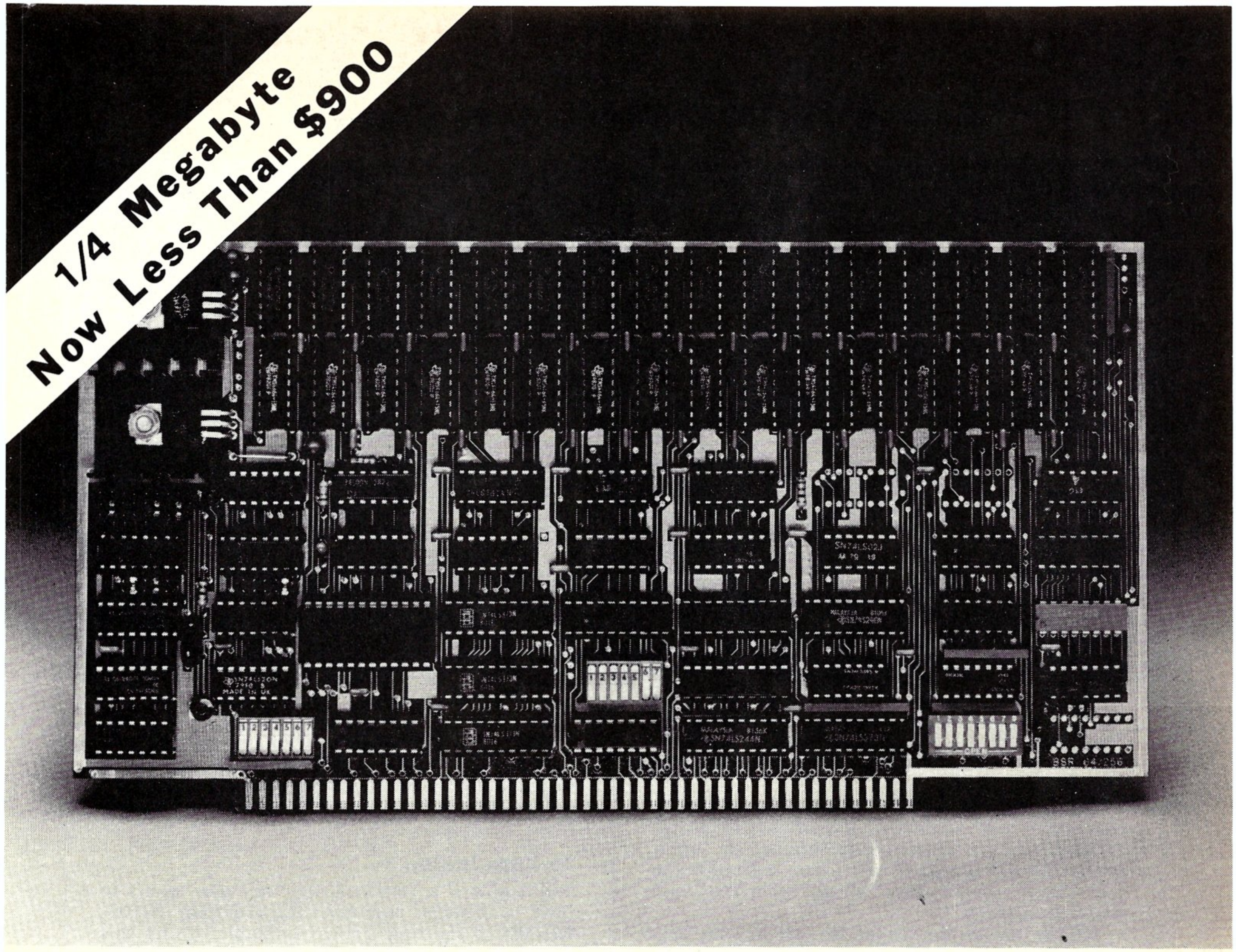
Board design

The Teleram card can hold either one or two Intel 7110 128K bubble memory modules. Thus one card can contain either 128K or 256K of bubble memory. Up to four cards can be used (in a daisy-chain) to provide up to 1Mbyte of storage. The average access time is 48 msec, and the maximum data rate is 100K bytes per second. The supporting circuitry allows bubble devices to be operated either in series (lower power, fixed throughput rate) or parallel (highest throughput rate), depending on user system requirements.

Data can be transferred in either a DMA (Direct Memory Access) or Polled mode. The board uses an Intel 7220-1 Bubble Memory Controller (BMC) and an Intel 8237-2 DMA Controller. The DMA is the fastest, allowing the 8237 to transfer data directly from the 7220 FIFO (first-in-first-out) RAM to or from system RAM. In the polled mode the CPU must periodically check the status register of the 7220 to determine if the BMC FIFO is ready to receive data or if the FIFO contains data to be read. The BMC may also generate an interrupt when its FIFO is either half full or half empty to indicate that data may be either written or read.

DMA operation should be used to provide the fastest operation. However, if timing problems are encountered, then the polled mode can be used. Teleram points out in their manual that "any CPU board which disables its own drivers during any DMA cycle may not function reliably with the bubble board."

Sol Libes, P.O. Box 1192, Mountainside, NJ 07092



PCE Systems' BSR 64/256 is an 8 bit bank selectable dynamic RAM card designed to operate in Z-80, 8080 and 8085 based S-100 computer systems with a bus master clock frequency of up to 4 MHz (A model) or 6 MHz (B model).

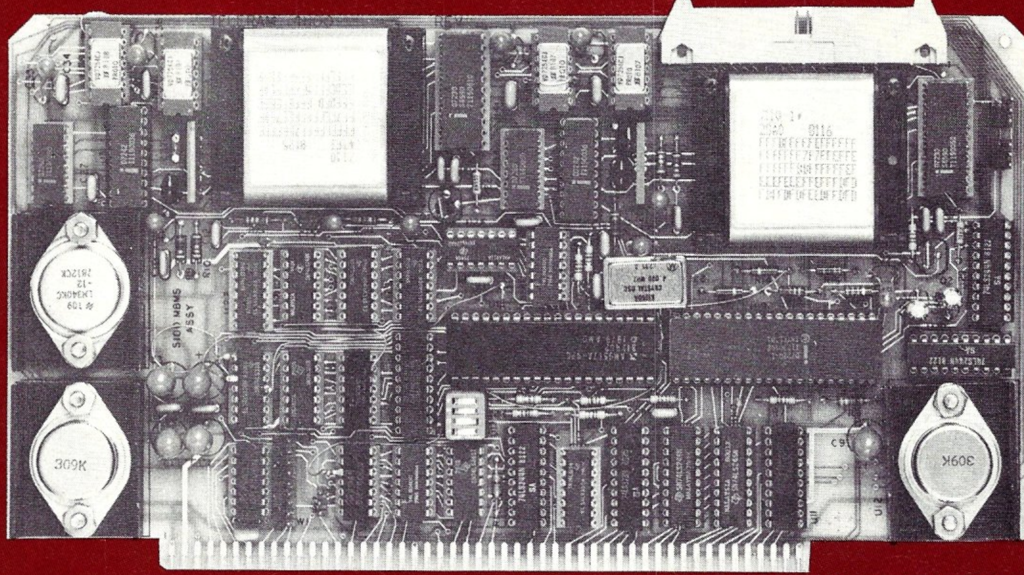
The BSR 64/256 uses the IEEE 696 8 bit address bus extension to select each individual bank of memory. If the host system is not capable of driving the extended address bus, one of the BSR 64/256 cards in the system may be configured to drive it through an onboard latched ouput port. Each bank of memory is configured as 16 independent software selectable 4K blocks. System area is allocated in 4K blocks by writing a system mask out to latched output ports. Another port allows any one of up to eight cards to be assigned as the current system master. Logically, up to 64 cards may be addressed in a single computer system. Features such as the following make the BSR 64/256 an incredible buy.

* Maximum storage of up to 256K bytes of memory per card and logically up to 64 cards per system.
* Bank selectable area is any combination of software selectable 4k blocks in any 64k bank.
* System area may be any combination of software selectable 4k blocks in low order bank of current system master card.
* Any one of up to 8 cards software selectable as system master; one card jumper selectable as system master on power-up or reset.
* Bank Selection uses or implements IEEE 696 (S-100) extended address bus.
* 4MHz (model A) or 6MHz (model B) operation with no wait states using invisible refresh.
* Needs only four consecutive I/O ports, selectable on any 4 port boundary, for entire system.
* Particularly suited for use with multi-user operating systems such as MP/M™* and MU.™**
* Attractive OEM and dealer pricing on all of our products.

PCE SYSTEMS
4219 S. Market Ct., Unit H
Sacramento, CA 95834
(916) 921-5454

* Digital Research ** PCE Systems

CIRCLE 107 ON READER SERVICE CARD



Teleram S-100 magnetic bubble memory card

PHOTO: TELERAM COMMUNICATIONS CORP.

The board includes circuitry to prevent data loss due to inadvertent power failure. Also error detection and correction circuitry is included, which can detect and correct burst errors up to 5 bits long per bubble page.

The board that we were loaned for preparing this review was obviously a preproduction board. It was nicely constructed but contained six jumpers on the wiring side of the board. Also only the LSI ICs were in sockets. All the other ICs were soldered in place.

DIP switches are included for selecting the I/O port addresses and DMA priority. Jumper pins are provided for selecting the interrupt vectors and setting system expansion. When more than one bubble memory board is used, only one need have the controller, DMA, and bus interface circuitry.

Software

Furnished with the unit was an 8″ SD diskette, which contained sample BIOS driver programs and utilities that allow the user to exercise and test the bubble memory system. BIOS driver programs for both polled and DMA transfers are provided. Altogether, the listings amount to about 1,000 lines of code and are well commented.

The utility program allows the user to determine if the board can be run in DMA or polled mode. It also allows the user to test the board for faults or marginal operation.

Documentation

Included with the board was a 37-page User's Manual, a copy of the Intel *Bubble Memory Prototype Kit User's Manual*, and a copy of the Intel spec sheet for the BMC (16 pages). The Teleram User's manual was marked "preliminary." It was a xerox copy and had several changes marked in it. The manual was very complete, containing theory of operation, software installation techniques, and a very complete set of professionally drawn schematic diagrams.

In conclusion

Right now if a personal computer user wants to have a bubble memory, the Teleram S-100-MBMS is the only choice available. Using this board, or up to four boards, a user can have a solid-state nonvolatile mass storage system ranging from 128Kbytes to 1Mbyte in size. It provides all the advantages of solid-state devices over electromechanical devices. The only disadvantage is the price.

Sol Libes is the editor of *Microsystems* magazine and the author of 15 books, including *Interfacing to S-100/IEEE-696 Microcomputers* (McGraw-Hill), *Digital Logic Circuits* (Hayden), *Small Computer Systems Handbook* (Hayden). He was the founder and served for many years as president of the Amateur Computer Group of New Jersey, one of the largest such organizations in the country; and is a co-organizer of SIG/M, the largest nonprofit distributor of public-domain CP/M-based software. These are extracurricular activities for Sol, who is a professor of electronic technology at Union County College, New Jersey.

Bubble memory systems have advantages over disk systems . . . where the system would be subjected to dust or mechanical shock, and in portable systems.

Teleram S-100 continued . . .

Product name:	S-100-MBMS Magnetic Bubble Memory System
Manufacturer:	*Teleram Communications Corporation* 2 Corporate Park Drive White Plains, NY 10604 (914) 694-9270 Telex 646658
Memory size:	128 Kbytes or 256 Kbytes per board; 1 Mbyte system maximum (4 boards)
Interface:	S-100; IEEE-696 compatible
Performance:	Rotating field rate: 50 KHz Data rate: 100 Kbytes/sec maximum Access time: 48 msec average
Electrical characteristics:	+8 VDC, 2A max. +16 VDC, 1A max.
Environment:	0-50°C
Price:	$1,945—256 Kbyte card $1,295—128 Kbyte card $1,860—256 Kbyte slave $1,210—128 Kbyte slave

HARDWARE REVIEW

The Jade Bus Probe

by Barry Press

The words "service" and "repair" are part of some obscure foreign language to many owners, users, and builders of S-100 microcomputer systems. Given the general lack of repair stations for S-100 computers, it's clear that they could never have grown so popular if it were not for the inherent reliability of LSI microprocessors. Unfortunately, the well-known theorems of Murphy prove that inevitably, no matter how reliable your system is, one of these days you're going to power up your computer and receive the electronic equivalent of "Huh?" Or, if you're the adventurous type, sooner or later you'll build a circuit that not only doesn't work, but crashes the rest of the system. Then what?

The really powerful tools for microprocessor testing are for the well funded—logic analyzers, in-circuit emulators, board testers, and marvelous things like that. There are painfully few alternatives for small shops and home experimenters—a scope, logic probes, DVM, and a pulse generator are about it. Trying to see more than one or two signals is hopeless, not to mention that all the cables lying about are more than enough to snare the unwary (or clumsy). Buses present their own problems too, such as which pin is the clock on, and is this pin 43 or 44? Or, tell me again how I get to those signals on the back of the board?

Pretty grim, right? Well, assuming you know how to service the machine, but simply lack enough tools, there's hope. The Bus Probe, by Jade Computer Products, 4901 West Rosencrans Ave., Hawthorne, CA 90250, is exactly what its name implies: a logic probe for the S-100 bus. It's like having 96 logic probes connected into the bus, without cables, and with some added features. All the S-100 bus signals (per IEEE-696, see the July 1979 issue of *IEEE Computer Magazine*) are brought out individually to an LED in a clearly labeled display. You can reserve your scope and ingenuity for the signals you really think are the problem, and can detect some common problems, like inoperative power supplies or bus lines shorted to +5 or ground, without a scope at all.

The Bus Probe is a 9" high, double-sided S-100 printed circuit card (Figure 1). The upper third of the card contains the LEDs in a 4 x 24 rectangular array. Each of the four rows of LEDs is arranged into three groups of eight. The circuit board behind the array is silk-screened to provide a black background for the LEDs, with the identification of each LED directly above it. Group labels appear directly above the individual labels. In operation, the combination of the silk screening and reasonably strong drive to the LEDs when triggered (about 20 ma) provide a bright, solid display.

To the right of the bottom row of LEDs is an eight-position DIP switch used to control which bus cycles are displayed. Below the LEDs and the switch is the circuitry. Most of this space is taken up by the integrated circuits used to drive the LEDs. The incoming bus lines are buffered to present only one LS-type bus load, and are then coupled into the display (Table 1).

Barry Press, 11921 Sonoma Way, Northridge, CA 91326

Table 1. LEDs grouped by functional category

Category	Bus probe signal name	IEEE signal name
Address lines	A0 - A23	A0 - A23
TMA	DMA0 - DMA3	DMA0 - DMA3
Disable	DOUT	DODSBx*
	STAT	SDSBx*
	ADDR	ADSBx*
	CNTL	CDSBx*
Data in	DI0 - DI7	DI0 - DI7
Data out	DO0 - DO7	DO0 - DO7
Vectored interrupt	VI0 - VI7	VI0 - VI7
Utility	INT	INT*
	NMI	NMI*
	PHAN	PHANTOM*
	HOLD	HOLD*
	PWF	PWRFAIL*
	ERR	ERROR*
	SXTN	SIXTN*
	MWRT	MWRT
Status	MEMR	sMEMR
	M1	sM1
	WO	sWO*
	IN	sINP
	OUT	sOUT
	XTRQ	sXTRQ*
	INTA	sINTA
	HALT	sHALTA
Clears	RESET	RESET
	SLAVE	SLAVE CLR*
	POC	POC*
Processor status	SYNC	pSYNC
	STVAL	pSTVAL*
	DBIN	pDBIN
	WR	pWR*
	HLDA	pHLDA
Wait controls	RDY	RDY
	XRDY	XRDY

Quite a bit of the circuitry area on the board is reserved for user customization of the board. Five spare IC sockets are provided, as well as six uncommitted inverters (an entire 74LS04) and two uncommitted retriggerable one-shots (an entire 74LS123). Additionally, a patch area (Figure 2) is provided to permit the signals from these circuits as well as elsewhere on the board to be used to control the display. Space is provided to mount two switches on the top of the board, with circuit traces provided to route the switch terminals down to the circuitry area.

Groups are also provided on the display for the power lines, clocks, display enable lines, as well as a group of eight uncommitted LEDs that may be user-defined through the patch area.

In a conventional logic probe, the LED simply responds to the signal at its input, with some of the better ones incorporating circuitry to indicate signals whose state is changing. Buses make the interpretation of signals difficult with such simple probes, since many different types of events occur in succession on the bus. For example, during the execution of a "MOV A,M" instruction by an 8080, an instruction fetch cycle will occur followed immediately by a data fetch. Systems incorporating direct memory access (DMA) may have DMA cycles interleaved among the cycles commanded by the processor. All these different events are simply jumbled together and reported by a conventional logic probe, making it nearly useless for bus analysis. Recognizing this problem, Jade has provided "enables" on each of the groups of LEDs. Enabling a group logically connects the LEDs to the bus lines: If enabled, the LEDs display the bus signals; if not, they are dark. By properly controlling the enable lines, it is possible to see only the instruction fetches, only the I/O outputs, etc.

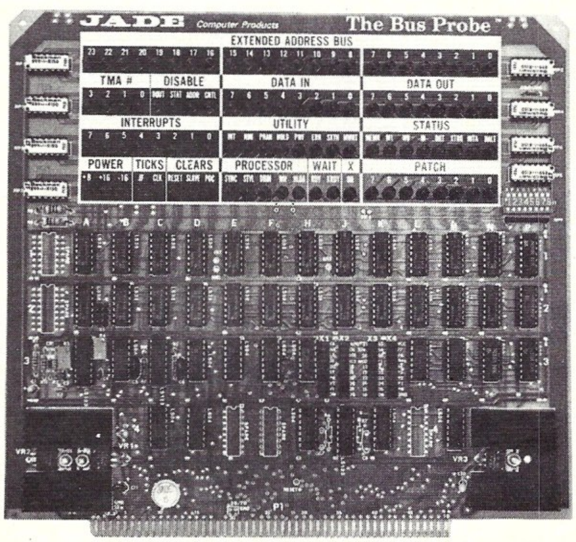


Figure 1. The Jade Bus Probe.

In the Bus Probe, enables are decoded from the bus. Enable signals are provided that become true for each of the following conditions:

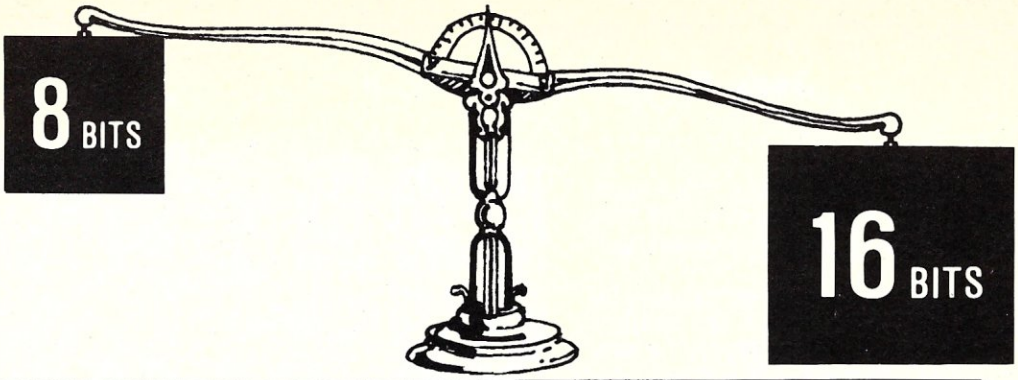
- An instruction fetch memory-read cycle
- A data fetch memory-read cycle
- A memory write cycle
- An I/O input cycle
- An I/O output cycle and
- Any bus cycle (i.e., always enabled)

It is also possible to couple a user-defined enable derived from logic in the patch area into the display. The enables in use at any given time may be selected with the DIP switch to the right of the display. One or more enable conditions may be selected; the display will be enabled when any of the selected conditions are true. In the standard configuration (i.e., before you modify the board), any active enable connects all the groups of LEDs to the bus.

The clock waveforms on the bus, 0, and CLOCK get special treatment from the bus probe. Since in normal operation the signals are oscillating, failures in which the signal was shorted to +5 (for example) would be difficult to detect if the signal were simply coupled to the LED directly. To solve this problem, Jade has coupled the signals to the LEDs through retriggerable one-shots. As long as the signals continue to oscillate, the one-shots will remain triggered, and the LEDs will remain lit. If a signal stops oscillating, either in the high or low state, the one-shot will expire, and the LED will go out. Simple.

Also on the board is an uncommitted oscillator, adjustable for pulse duration and frequency. The oscillator may be connected to the reset line for examination of the reset behavior of the system, or to one of two patch points. The ability to generate repetitive reset (or other) pulses is particularly useful with a scope, such as for measuring the length of the bus reset pulse. In my system, for example, too many boards from too many vendors are unnecessarily (and incorrectly) pulling up the reset line, causing the system to reset improperly at power up. Problems like that are easy to find with the Bus Probe and a scope. Maybe someday I'll even fix the offending board instead of holding down the reset button.

The Bus Probe makes possible a reasonably straightforward approach to repairing a failed system. I'll assume that you have, in addition to the Bus Probe, a triggered oscilloscope, an extender card, and a good monitor program in the CPU (i.e., one that will permit you to store into and dump memory, to jump to a program, and to read and write I/O ports). The first step is to try and get the CPU operational enough to use the monitor. The method is to remove all the cards from the mainframe except those required to run the monitor, and insert the Bus Probe into an available slot. I'd suggest one near the front, leaving enough room for the extender card in front of that. (Putting it at the back creates the problem of having to route all the cables from the back panel around it on their way to the circuit cards.) On systems with enough memory on the CPU card to support the monitor, and with a terminal also connected through the CPU, this may simply be the CPU



MICROCOMPUTERS

THE CHOICE JUST GOT EASIER!

PRICE $4390 Gets you a Tec-86 Complete 16 Bit 8086 System, including 64K Bytes of Memory, Dual 8 Inch Double Density Floppy Disks, and more. . . (See complete Tec-86 specs below.)

PERFORMANCE Faster Processing, Increased Address Space, up to One Megabyte of available memory, and the Increased throughput of the the TEC-86 16 Bit System means that you can now do work on a microcomputer never before practical with the slower 8 Bit systems.

SOFTWARE CP/M-86™ Brings the World of CP/M software up to the 16 bit systems. BASIC-86™, MS-Fortran™, MS-PASCAL™ are just a few of the Languages currently available.

EXPANSION Memory Expansion up to ONE MEGABYTE. Disk Storage can be increased to over 30 Megabytes with our Winchester Disk Subsystem. Multi User Systems available soon!

QUALITY Full ONE YEAR WARRANTY is standard on TECMAR components. This Warranty is possible due to our High Engineering Standards, and Thorough Product Testing.

DELIVERY We have been delivering 16 Bit Boards and Systems for nearly two years. Most orders are shipped from stock or within 30 days.

S100 A/D $765
Analog to Digital Converter with Timer/Counter

12 Bit Accuracy, 16 Channels, 30 KHz Conversion, Five Programmable Timer/Counters for:
- Time of Day
- Triggering Conversions
- Counting Conversions
- Counting External Signals
- Interrupting the CPU

Options: Programmable Gain; 14, 16 Bit Accuracy; 40, 100, 125 KHz Conversion; Expansion to 256 Channels

$4390 TEC-86 $4390
16 BIT 8086 MICROCOMPUTER SYSTEM

FEATURES: 5 MHz 8086 CPU * 64K Bytes RAM * S100 IEEE-696 * Dual 8 Inch Double Density Floppy Disk Drives * Two RS 232 Serial Ports, 50 to 19200 Baud * Three Parallel Ports * ROM Boot for CP/M-86 * Heavy Duty Power Supply * Attractive Industrial Quality Desk Top Cabinet * Completely Assembled and Tested * **OPTIONS:** Up to One Megabyte of Memory, Double Sided Double Density Floppy Disks * Up to 31 Megabytes of Winchester Fixed Disk Storage. * CP/M-86 * BASIC-86 * MS-FORTRAN * MS-PASCAL * CBASIC/86™ * PASCAL/M86™

$10990 TEC-86W $10990

Same Great Features as our TEC-86 except that the TEC-86W has 256K of RAM, and our 31 Megabyte Winchester Disk Subsystem.

NEW — 64K/256K S100 MEMORY $595/$1695

8 and 16 Bit Transfer for 8 and 16 Bit Systems
MP/M and MP/M-86 Compatible, Up to 7 users on one 256K board.
Parity Detection with Interrupt and LED Indication
Hardware Write Protect
Full DMA Capability
24/20/16 bit Address Decoding
Enable/Disable 8K/32K Segments
Bank Select in 8K/32K Segments
Meets IEEE-696 Specifications

See our catalog for many other fine S-100 and Apple Boards including: — Analog to Digital Converters *(16-256 channels; 12, 14, or 16 bit accuracy; 30, 40, 100, or 125 KHz; programmable gain; timer/counters)* **— Digital to Analog Converters** *(12 bit accuracy, 3 microsecond conversion)* **— 8086 CPU Board — I/O Boards — 16K Static RAM — Real-time Video Digitizer and Display.**

TECMAR INC.

DEALER INQUIRIES INVITED

23600 Mercantile Rd. • Cleveland, OH 44122

TECMAR, INC.
(216) 464-7410

CIRCLE 95 ON READER SERVICE CARD

itself. In systems such as mine with a video card, PROM card, and memory card, this comes to four or more cards, depending on the memory configuration, etc.

Having set that up, power on the computer and see what happens, such as whether the monitor runs. If not, first check for power supply failure. The LEDs on the Bus Probe will tell you if a supply is completely dead or shorted; your scope can tell you if the voltages are within tolerance, and will check the output of the onboard regulators. If the supplies are within tolerance and it's a single-board computer supporting the monitor, then switch on the "always on" enable on the Bus Probe, and see what's there. You should certainly have at least one clock signal; the other signals to be expected will depend on the board and the monitor. Look particularly for unexpected behavior on some of the lines not used for simple operations, such as the interrupt or DMA lines. If bus cycles are visible, see if some of the essential control lines or the data lines appear to be stuck high or low. Problems could be caused by a failure within the CPU itself, by defective bus drivers, or by a short on the motherboard.

If more than just the CPU has to run to support the monitor, do the basic checks above first. If the problem isn't visible, then see if the reset pulses are getting distributed properly, and if they have sufficient duration to reset all the dependent circuits. The reset line pulser on the Bus Probe plus your scope makes this easy. If the resets are there, use the more selective enables on the Bus Probe, such as selecting only instruction fetch cycles for display, to try and see what the processor is trying to do. Setting the reset pulser for (relatively) long intervals between pulses may help by isolating just the first few instructions on the display and for your scope. The switches on the top of the Bus Probe could be wired to cause single-stepping, possibly by clocking a flip-flop to release a wait state. The flip-flop would be set to generate the wait state by the sM1 signal. Single-stepping would be enabled by one of the switches, and steps initiated by the other. Single-stepping after a reset, for example, while following along in your listing of the monitor and verifying the instructions fetched and data transferred, should help immensely in localizing the problem. Be careful in designing your single-step logic, however, to ensure that you don't interfere with dynamic memory refresh or other processes.

Once you get your monitor running, you can use the computer to help diagnose itself. The general idea is, once you have an hypothesis about what's wrong, set up a program to exercise only that condition, and monitor the results on the bus with the Bus Probe. For instance, I recently had a large cabinet full of books fall on my computer, causing

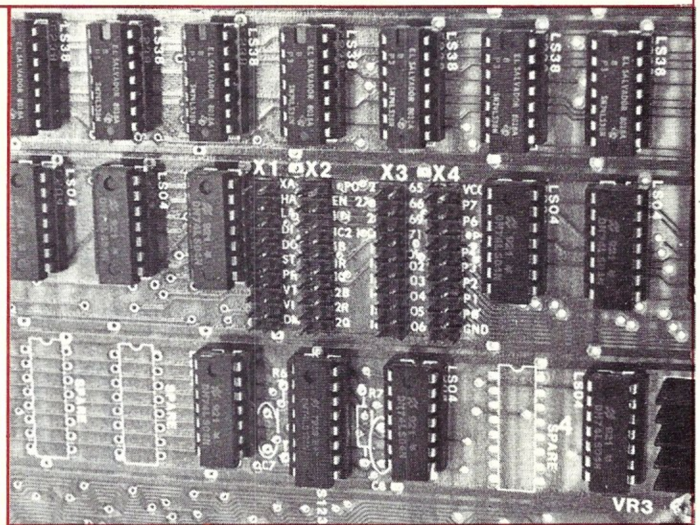


Figure 2. Patch area of Jade Bus Probe.

the printer to emit garbage instead of the eloquence I'd expected. A quick check with a second Bus Probe (the first test unit being another victim of falling technology) revealed that the right data was being sent to the parallel port, showing that the problem was either in the port circuitry, the cable, or the printer itself. As a more complex example, suppose the problem in your system seems to be that the disks are inoperative. After checking the obvious and simple things (power on, supplies working, cables plugged in, etc.), you conclude that the problem is that the CPU and the controller just aren't on speaking terms. In my system, the CPU determines status from the controller through an input port. To investigate the problem, I'd set up a program that kept trying to activate the disk controller, and would set up the Bus Probe to monitor the bus only on I/O inputs (remember the enables?). By watching the status port on the Bus Probe, combined with prowling around the disk controller with my scope, I should be able to track the problem down.

At $119.95 for the kit and $149.95 assembled and tested, the Bus Probe is comparable in price to a low-cost Digital Volt Meter (DVM), and (as a kit) about the same price as three hand-held logic probes. I think that's a good value. Although the unit I reviewed was already assembled, I have built the Double D disk controller by the same people, and found the assembly instructions well written and easy to follow. Examination of the Bus Probe manual revealed it to be equally well written, including an extensive series of tests to verify correct operation.

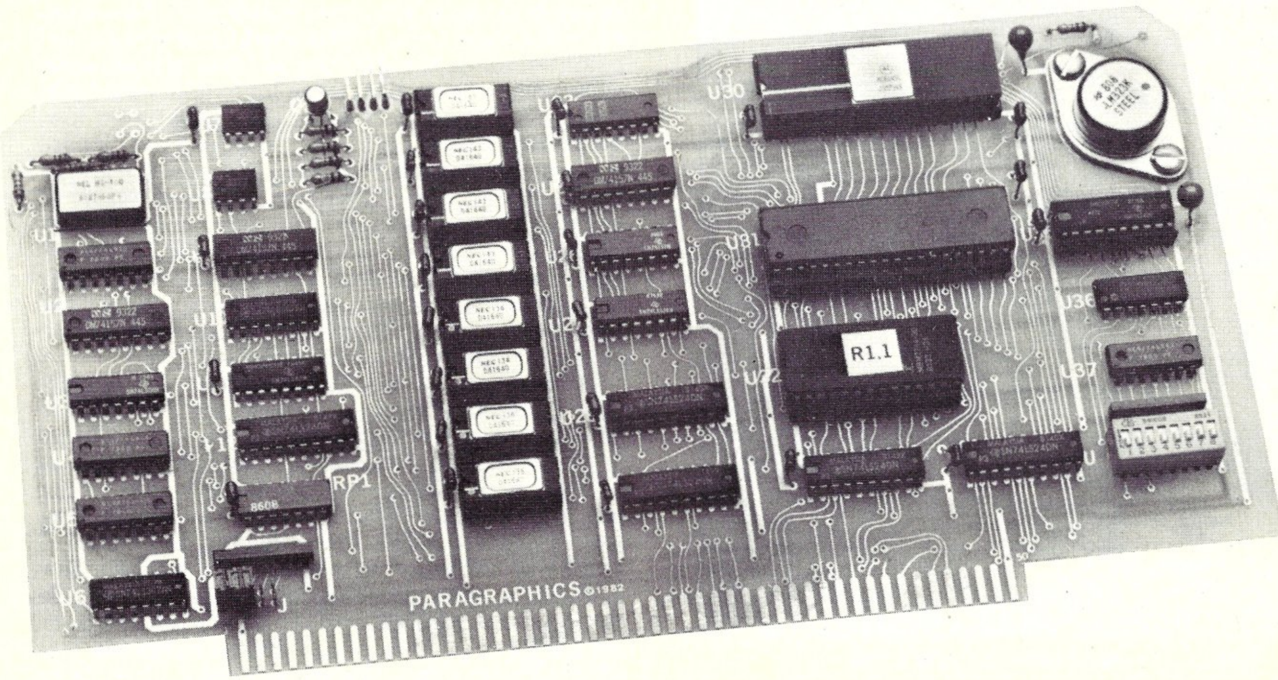
To me, the most significant disadvantage of the Bus Probe is that it has no provision to latch signals. The enables directly connect the LEDs to the

The Bus Probe makes possible a reasonably straightforward approach to repairing a failed system.

Introducing

"THE GAME BOARD"

by paraGraphics



S100 SINGLE BOARD GRAPHICS COMPUTER

introductory price $295 *

"The Game Board" from paraGraphics offers high quality, high density monochromatic graphics at an affordable price. Combined with your S100 computer and a video monitor (composite or direct drive), "The Game Board" offers an economical solution to the high cost of computer graphics.

Features:

- State of the art 64k RAM technology
- User selectable resolution
  512 x 576   512 x 288
- High level graphics commands
- Allows fast manipulation of user generated sprites.
- Terminal emulation mode executes a subset of the H19/Z19 command set
- User selectable terminal density
  85 x 24   85 x 48
- Synchronous processor provides fast flicker free display

Available Soon:

- Fully assembled and tested
- Graphics applications packages
- Games

* **"The Game Board"** is offered as a partial kit including blank p.c. board, 8k graphics operating system PROM, address decoder ROM, crystal clock generator, full documentation, and sample programs.

For more information and ordering contact:

paraGraphics
P.O. Box 67
South Easton, Mass. 02375
(617) 620-4513

VISA   MasterCard

Jade Bus Probe continued . . .

bus, permitting them to display whatever data used to latch the bus contents into a register for display, so transient, nonrepeating events won't show up. If you're up to a little simple digital design, however, you can fix this problem yourself. For instance, two 74LS77 transparent latches inserted into two of the spare positions and connected to the eight spare LEDs would give you the capabilitiy to view eight bits of data dynamically when their enable lines were high, and would statically freeze the data when the enable went low. Some of the other spare IC positions could be used to set up the circuits necessary to derive the proper enable signal. Alternatively, a couple of 74LS174 hex flip-flops could be used in the same way, freezing the data present on the leading edge of an enable. This ability to customize the board for your specific needs was intentionally designed into the Bus Probe, and virtually guarantees that it can help solve the most troublesome problem.

Additionally, I would have liked the connections of the enable lines to the groups of LEDs to have a little more flexibility. For instance, the individual enables decoded on the board, such as I/O input or output, are not brought out to the patch area separately. Consequently, connecting some groups of LEDs to one enable, but different groups of LEDs to other enables, is difficult. For instance, a likely combination would be to enable the data in LEDs on an I/O input, and the data out LEDs on an I/O output. This simply isn't possible without some delicate soldering of wires to integrated circuit pins.

The Bus Probe won't find all your problems by itself, and it certainly won't do you any good if you don't know how to service digital electronics in the first place. But a lot of you have the necessary skills, and are simply lacking the tools. Using the Bus Probe in conjunction with an inexpensive triggered sweep oscilloscope, the hobbyist or small shop can have a fighting chance at servicing S-100 computers for under $500, and will also find it useful for integrating new peripherals (for instance, scrambled bits from that new keyboard are immediately apparent). Rather than simply build yet another CPU or memory board, Jade has applied some creative thought to a serious problem and has come up with a valuable product at a good price. We can all hope Jade continues in the same vein, perhaps with a card to permit an S-100 computer to perform as a logic analyzer, or with an overgrown parallel output card able to act as a programmable waveform generator. I applaud their efforts, and encourage you to investigate the Bus Probe yourself.

Barry Press has a B.S.E.E. and a Master's degree in computer sciences. He is a Vice President of Compunet, Inc., in charge of Microprocessor Products, and is currently concerned with the development of real-time microcomputers and a turnkey microcomputer system for doctors.

8086/WINCHESTER DISK DRIVERS

Use your Seattle Computer Products System 1 or System 2 to its full potential with Morrow Designs M10, M20 or M26 Disk Subsystems on the most popular operating systems.

Driver for
:*: 86-DOS Version 1.25 — specify M10 or M20 or M26 (with floppy) Driver.
$300.00

Driver for
:*: CP/M-86 Version 1.1 — Requires System 1 or 2 with double sided floppy disks and/or M10 or M20 or M26 disk subsystem. Specify version when ordering.
$200.00

Driver for
:*: MP/M-86 Version 2.0 — Requires System 2 (128k ram) with double sided floppy disks, SCP 4-Port Serial Card and/or Discus M10 or M20 or M26. Specify version when ordering.
$300.00

A. CP/M-86 and MP/M-86 are not included. Order separately from E.D.S.

B. All drivers contain a utility to format their respective disk drives.

:*: CP/M-86 and MP/M-86 are registered trademarks of Digital Research Corporation.

:*::*: 86-DOS is a registered trademark of Seattle Computer Products.

DEALER & DISTRIBUTOR INQUIRIES ARE INVITED

Complete systems and subsystems are also available.

E.D.S. PRODUCTS DEALERS LIST:

Eastern:	Western:	Europe:
CDR Systems, Inc.	S-100 Inc.	Micro Standard AB
437 College Park Dr.	14425 North 79th St.	Dobelnsgatan 95
Monroeville, Pa. 15146	Scottsdale, Az. 85260	S 113 52 Stockholm
(412) 325-1708	(602) 991-7870	Sweden

EXECUTIVE DATA SYSTEMS INCORPORATED

post office box 205
roselle park, new jersey 07204
(201) 241-4887
EST. 1979

CIRCLE 40 ON READER SERVICE CARD

KEY SYSTEM 16

5795

A True 16 Bit S-100 Computer

Available NOW

assembled & tested as a unit using Compupro boards

With:
- CPU 8086 (10 Mhz)
- 128k memory (16 bit wide)
- 3 serial 2 parallel I/O ports
- DMA Floppy Controller
- 2 DD DS 8" Floppy Drives in enclosure with power supply and cables
- 20 slot S-100 Enclosure
- CP/M 86

KEY SYSTEM 16H10

$8190.00

as above except, instead of 2 floppies, system includes 1 8" DD DS Floppy, and one 10 Mb 8" hard disk.

KEY

MICRO SYSTEMS INC.

822 Boylston St., Chestnut Hill, MA 02167

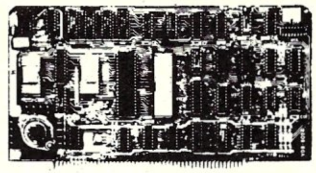
(617) 738-7308

Compupro is a trademark of Godbout Electronics
CP/M 86 is a trademark of Digital Research Corporation

CIRCLE 113 ON READER SERVICE CARD

*Dirt Cheap**

SDSYSTEMS™

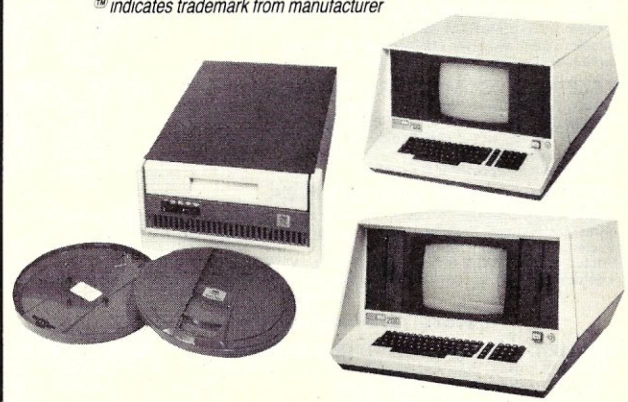


S-100 Boards

components and systems

MPC-4 — MULTI-PORT COMM.	**$495**
△ VERSAFLOPPY II	**350**
△ EXPANDORAM II — 64K	**417**
△ SBC-200 w/Monitor	**299**
△ CP/M™	**225**
Z-80 Starter Kit	**275**
VDB 8024	**465**
△ ***SPECIAL*** — VFY II, EXP II, SBC 200 —	**1050**
w/CP/M™	**1250**

prices subject to change
™ indicates trademark from manufacturer



Complete line of SD boards, components and systems at below market prices. We offer the latest factory releases and complete technical support.

Free brochures available upon request.

***If we can't offer you the lowest prices . . . we'll tell you who can.**

MCC MICRO COMPUTER COMPANY INC.

301-942-5442

701 Wheaton Plaza North
Wheaton, Maryland
20902

CIRCLE 109 ON READER SERVICE CARD

LOOK NO FURTHER.................................. WE'VE GOT YOUR PRICE!

ALTOS COMPUTERS

	RETAIL	PREPAY
5000-15 Single User (64K, 2-1MB 5" disks)	$2995	$2399
5000-5D Single User (64K, 1MB + 5MB 5" disk)	$5995	$4799
8600-10 8-User (512K, .5MB + 10MB disks)	$10500	$8399

AVL 'EAGLE' SERIES DESKTOP COMPUTERS

Eagle II Desktop System (64K, dual 400K disk)		$2499
Eagle III Desktop System (64K, dual 800K disk)		$3299
Eagle IV Desktop System (64K, 800K + 7 .5MB disk)		$4999
for 'Acctg. Plus' & 'Spellbinder' W/P ADD		$1800

COLUMBIA DATA PRODUCTS COMPUTERS

	PREPAY
NEW! Multi Personal Computer (128K, 2-400K disks, 8-slots) (IBM PC LOOK-ALIKE - far greater value)	$2995
1503-2 System (128K, dual 800K disks)	$2845
1802-1 System (128K, dual 1MB 8" disks)	CALL!
CONCEPT 1000 - *MultiBus* - up to 16 users	CALL!

CORVUS DESKTOP COMPUTERS

32-bit Mainframe power in a desktop system...

NEW! 'Concept' (256K, 68000, 120X56 CRT) from	$4995
Disk subsystems (6, 11 & 20MB) from	$2695

DIGITAL EQUIPMENT CORP. (DEC)

Destined to become legends in their own time...

	PREPAY
'Rainbow 100' Z80/8088 w/CP/M-80/86, 132X24CRT	$3495
Professional 300 series (F11 CPU) from	$3995

DYNABYTE S100 COMPUTERS

	RETAIL	PREPAY
NEW! 'Monarch' 6600 (256K, 10MB Disk, etc.)	$10995	$9995
NEW! 'Monarch' 6900	$11995	$10895

MORROW 'DECISION' COMPUTERS

	RETAIL	PREPAY
Micro Decision (64K, dual 200K disks, Word Stat, etc.)		$1495
'Desktop' model (65K, 2-400K 5" disks)	$4195	$3495
'Rackmount' model (65K, 2-500K 8" disks)	$4375	$3695
Discus M10 (10MB 5" rigid disk subsystem)	$2795	$2495
Discus M16 (16MB, 5" rigid disk subsystem)	$3495	$2995
Discus M20 (20MB 8" rigid disk subsystem)	$4795	$3995
A822S Dual Slimline 8" floppy pkg. (2MB)	$1995	$1695

N.E.C. DESKTOP COMPUTERS

Hold on to your hats . . . the 'Advanced Personal Computer' has arrived! **Desktop computing will never be the same! (IBM DATA MASTER LOOK ALIKE)**

	RETAIL	PREPAY
NEW! 'HO2' 128K, 8086, 2MB 8" disk GREEN screen		$3995
NEW! 'HO3' 128K, 8086, 2MB 8" disks COLOR		$4995
Options: 128K RAM **$600**		
Graphics from $600, 8087, 2S + 2P I/O		$250

SANYO DESKTOP COMPUTERS

	RETAIL	PREPAY
MBC-1000 'Creative Computer' 66K, Z80, 300K disk	$1995	$1695
BDF-160 Single 300K Add-on Drive Pkg.	$995	$895
BDF-360 Dual 300K Add-on Drive Pkg.	$1595	$1339
BDF-860 Dual 1MB 8" Add-on Drive Pkg.	$2795	$2395
MBC-2000 64K Desktop System (2-300K disk)	$3495	$2995
MBC-3000 64K Desktop System (2-1MB disk)	$4995	CALL!

SYSTEMS GROUP S100 COMPUTERS

	RETAIL	PREPAY
System 2814 (64K RAM, 2MB Disk)	$5890	$4599
System 2824 Two User (128K RAM, 2MB Disk)	$6980	$5449
System 2819 (64K RAM, 11MB Disk)	$8905	$6949

TELEVIDEO COMPUTER SYSTEMS

	RETAIL	PREPAY
TS-800 64K Satellite Wrkstn. for TS-806	$1795	$1399
TS-802 Desktop System (64K, 2-800K disks)	$3495	$2699
TS-802H Similar with 7.5MB Rigid Disk	$6995	$5299
TS-806 Computer (64K, 800K + 7.5MB disk)	$6995	$5499
'TeleSolutions' (WordStar & CalcStar)	$495	$400

VECTOR GRAPHIC S100 SYSTEMS

Vector offers ON-SITE service & the most powerful W/P & spreadsheet software available on CP/M systems. TOP RATED in user surveys for over three years!

System 2800 - dual 1MB 8" drives	IN STOCK!	$5495
Vector 4/20 (128K, 2-600K disks, Z80B + 8088)		$4495
Vector 4/30 (128K, 600K + 5MB disks, Z80B + 8088) With Memorite III Included		$5995

If you can find a better value than Vector - BUY IT!

XEROX DESKTOP COMPUTER SYSTEMS

	RETAIL	PREPAY
'SAM' 820-1 (64K, dual 92K drives)	$2995	$2395
'SAM' 820-2 (64K, dual 8" 256K drives)	$3795	$2995
'SAM' 820-II (64K, dual 8" 1MB drives)		CALL!

Add **$169** for CP/M

VIDEO DISPLAY TERMINALS

	RETAIL	PREPAY
ADDS 'Viewpoint' series from..............................	$695	$589
ADDS 'Viewpoint 60' emulating terminal	$895	$799
AMPEX Dialog-80 **CLOSEOUT SPECIAL**	$1195	$699
DEC VT-101	$1650	$1099
DEC VT-131	$2450	$1669
DataGraphix 132-1 Hacker Specials from..............................	$2000	$299
TeleVideo 910	$699	$589
TeleVideo 925C	$995	$799
TeleVideo 950C	$1195	$949
NEW! TeleVideo 970		CALL!
NEW! TeleVideo 'Intelligent I' (w/64K RAM)	**$1695**	**$1399**
Wyse WY-100	**$995**	**$899**
Wyse WY-200 (132x25)		$1199

MATRIX PRINTERS

	RETAIL	PREPAY
NEW! Anadex DP9501A Printer (200 cps)	$1725	$1399
NEW! Anadex DP9620A (300/150 cps) DP/WP Printer		$1599
DEC Letterprinter 100 (30/80/240 cps) DP/WP Prtr.		$2699
Florida Data OSP120 (600/150 cps) Printer	$3900	$3399
Flor. Data OSP120 (600/150 cps) printer	$4200	$3699
ID'S 'Prism' 4-color printer (150/200 cps)	$1995	$1699
IDS Micro Prism 480 Printer	$795	$649
Tally MT160-I series (40/160 cps) from..............................	$845	CALL!
Tally MT1800 series (50/200 cps) Printers	$1995	$1639
Okidata 82A (120 cps - RS232 & Cent.)	$649	$499
Okidata 83A (120 cps - wide carrage)	$995	$799
Okidata 84 (200 cps - Dot Plot Graphics)	$1495	$1199

LETTER QUALITY PRINTERS

	RETAIL	PREPAY
Diablo 620 Printer (25 cps)	$1695	$1399
Diablo 630 Printer (40 cps)	$2900	$2099
Diablo 630 KSR Terminal (40 cps)	$3325	$2749
NEC 3510 (RS232(or 3530 (Cent.) Printer	$2295	$1899
NEC 7700 Printer (O.E.M. parallel)	$2995	$2399
NEC 7710 (RS232) or 7730 (Cent.) Printer	$3095	$2499
NEC 7720 KSR Terminal (55 cps)	$3610	$2899
Qume Sprint 9/35 (35 cps) Printer	$2160	$1899
Qume Sprint 9/45 (45 cps) Ltd. Panel	$2495	$2099
Qume Sprint II		CALL!
TEC (C. Itoh) F10 Printer (40 cps)	$1995	$1499
Single Bin Sheet Feed or Envelope Feed Mechanism	$1550	$1100
Dual Bin Sheet Feeder Mechanism	$2300	$1600
Envelope Feeder Mechanism	$1495	$1100
(Please specify NEC, Qume, Diablo or TEC model)		
CLOSEOUT - Qume Sprint 3 'SpeedFeed'	$1295	$399

SOFTWARE 'BEST-SELLER' LIST

	RETAIL	PREPAY
Ashton-Tate 'dBASE II' Relational DBMS	$700	$530
'dBASE II' for 8088/8086		CALL!
Aspen 'ProofReader & Grammatik'	$279	$229
BPI Acctg., Job Cost or Inventory Modules	$395	$319
CIS COBOL Compiler	$850	$750
Digital Research PL/1-80	$500	$425
Digital Research 'Pascal MT+'	$475	$425
DMA 'THE FORMULA' Applications Generator	$595	$500
Ecosoft 'MicroStat' Statistics Library	$295	$269
Fox & Geller 'Quickscreen' Screen Bldr.	$149	$135
InterSystems 'PASCAL-ZB' Z80 PASCAL	$395	$350
Lexisoft 'Spellbinder' Word Processing	$495	$300
MarketFax Sales & Marketing Tools	$500	$425
mDBS III with DRS, QRS & RTL Utilities		$3000
MicroPro WordStar w/ MailMerge *& Tutorial*	$665	$429
MicroPro 'DataStar' File Manager	$350	$235
Microsoft BASIC-80 Compiler	$395	$325
Microsoft FORTRAN-80 Compiler	$500	$400
Microsoft COBOL-80 Compiler	$750	$600
NECHE 'Select' or NECIS 'Benchmark' W/P	$495	$400
NECHE 'Report Manager' Electronic Wksht.	$200	$169
NWA 'Statpak' Statistics Library	$495	$400
NPI Law Office Mgr. w/ G/L & Docket Control		$4000
Oasis 'The World Plus' Speller (50K words)	$150	$109
Open Systems Acctg. & Inv. (per module)	$995	$795
Palantir Word Processing Package	$425	$350
Peachtree 'PeachPak 4' (GL, AR, AP, PR, INV)	$2475	$2000
'Reformatter' CP/M to IBM -or- DEC formats		$200
'Select' w/'SuperCalc' W/P Package	$595	$400
Sorcim 'SuperCalc' Electronic Worksheet	$295	$219
Sorcim 'Pascal/M' for 8086 and 8088	$495	$400
Sorcim 'TRANS-86' Translator for 8086	$125	$109
Sorcim 'ACT-86/88' (8086/8088 X-assembler)	$175	$139
Star Systems Acctg. (GL, AR, OE, AP, PO, PR)	$1600	$1100
Star Systems Property Management	$950	$799
Systems Plus 'Acctg. Plus' (8 modules)	$4760	$3500
Systems Plus (Any One Module)	$595	$469
Target PlannerCalc gridsheet		$50
TLB Solomon I Acctg. System (MDBS based)	$2595	$2200
TLB Solomon Series II Job Cost Acctg.	$3495	$3000
Vector 'VPS' Series (MEM3, EPL2, etc.)	$1585	$1250
Vector 'CONECT' Communications Software		$150
VectorVector CP/M2.5 128K Upgrade (w/64K RAM card)		$895

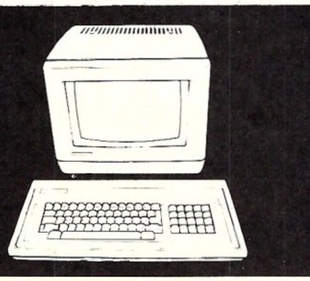
(much more available - please call)

IMB offers the FINEST SELECTION and **BEST VALUES** in computers, peripherals and software, for the discriminating buyer. Our affiliate, Budget InfoSystems, is an 'Authorized Distributor' for many product lines and offers 'Full Service' for each - **COMPLETE WARRANTY PROTECTION IS ALWAYS ASSURED!**

IBM specializes in service to **FOREIGN, PROFESSIONAL, MILITARY, GOVERNMENTAL, INSTITUTIONAL, and O.E.M.** markets! Call for details regarding quantity discounts and low cost leasing plans.

IBM values all of its customers, but we must encourage the untutored novice to purchase from a local '*Authorized Dealer.*' to preserve product integrity. Feel free to contact our office if no local dealer is available. We must insist that 'Turn-key Systems' be purchased through our 'Full Service' associate. **Budget InfoSystems,** to assure proper customer support.

TERMS: PREPAYMENT — **CONTACT**; Scott Secor, Gen'l Mgr.
F.O.B.: MPLS MN — **Delivery**: Stock to 4 wks. A.R.O.



International Microcomputer Brokers
607 NE Highway Ten—Jefferson Center
(across from NorthTown Mall)
Blaine, MN 55434
(612) 780-5361 -or- 786-5545
Hours: Daily 10AM to 6PM CST

CIRCLE 111 ON READER SERVICE CARD

Software Directory

Program name: ALIST, Alphabetical LIST and database program
Hardware system: 8080/Z80 CP/M
Minimum memory size: 48K
Language: Machine code
Description: Alphabetically ordered records of variable length can be easily maintained by ALIST. Buffering allows the entry of 50 records without a disk wait. Records can easily be added, corrected or deleted. ALIST files can be re-sorted by any field. Powerful selection conditions can be applied to combinations of fields. Selected records can be sent to the screen, the printer, new ALIST files, or CP/M text files (merge format optional). The user can create his own file systems and design his own print formats. ALIST is ideal for FILE CARD systems, REFERENCE/CITATION systems, MAIL/PHONE LISTS and APPOINTMENT files.
Release: September 1982
Price: $150; Manual, $25 (refundable with purchase)
Included with price: ALIST and file setup program on SSSD 8″ (or 4.25″ Osborne or SuperBrain) disk, Manual, on-line INDEX file, and MAIL LIST/LABEL demo program.
Where to purchase it:
Honor System Software
2562 East Glade
Mesa, AZ 85204
(602) 892-2434

Program name: DO
Hardware system: Standard 8″ CP/M or Superbrain
Minimum memory size: Any CP/M system
Language: Assembler
Description: DO.COM allows CP/M operator to type a string of CP/M commands that will execute one at a time. Allows typing ahead and leaving console while commands execute. No need to sit and wait for a command to complete so that next can be type in.
Release: October 1981
Price: $29 & tax in N.Y.
Included with price: 8080 source file on 8″ disk or Superbrain
Where to purchase it:
Stok Computer Interface
P.O. Box 501
Woodside, N.Y. 11377
(212) 476-7022

Program name: PLAN 1040
Hardware system: S-100, TRS-ModII, IBM or APPLE with CP/M
Minimum memory size: 56K
Language: MBasic 5.0
escription: A professional quality tax planning model for individual federal taxation. Runs five simultaneous alternative tax consequences to simulate various contemplated transactions. By varying combinations of input assumptions, user can instantaneously simulate impact of unlimited number of plans and get answers to any of his "What if" questions. Plan 1040 provides one comprehensive tax summary report and six supporting detail reports with complete audit trail for the computations. Regular Tax, Alt. Min. Tax, Income Avg. Tax, Min. Tax, Capital Gains/Losses, Lowest Tax, Inv. Tax credit limits, and effective tax rates are detailed in these reports. Completely menu driven and requires no programming or systems knowledge by user. Programs contain latest tax laws and will be updated whenever changes occur.

The mainframe version of this model has been used nationwide through Time-share for past 9 years.
Release: July 1982
Price: $295
Included with price: Distribution disk and manual
Where to purchase it:
Success Management Consul-tants
318 Surfview Drive
Pacific Palisades, CA. 90272
(213) 454-8030; 454-2624

Program name: Dual Tasking Forth
Hardware system: Z80 CP/M or CDOS System
Minimum memory size: 32K
Language: Object code
Description: Dual Tasking Forth permits simultaneous execution of two programs. User may write a Forth program, run it, and then continue to use his Forth system for other tasks. Normal program development activities may be continued while background task is executing. Two different programs may be run concurrently. Foreground task may be used to monitor and control background task.

No interrupts or real-time clocks are required, although they may be used if desired. The system produces a pseudo real-time clock value that may be used for event synchronization or other purposes. Less than 10% of processor time is devoted to Dual Tasking function. No extra memory is required. The system is easy to use, requiring little more than writing a program and telling system to run it in background. Two demonstration programs are included. All facilities of Timin Forth release 3 are included, such as Visual Editor and CP/M utility package. (Timin Forth release 3 is a superset of FIG Forth with many enhancements.)
Release: July 1982
Price: $285
Where to purchase it:
Timin Engineering Co.
6004 Erlanger St.
San Diego, CA 92122
(714) 455-9008

Program name: SECURE
Hardware system Any Z80-based CP/M 2.X system
Minimum memory size: No requirement
Language: Z80 object code
Description: SECURE will stop unauthorized access to any kind of CP/M file whether machine code, data, or text files. The program encrypts files using user-specified "keys" and can be used repeatedly on the same file using additional, different keys. It is ideal for use on mailing lists, financial data, and programs.

Software Directory continued . . .

Release: January 1982
Price: $150
Included with price: User manual and disk
Where to purchase it:
Century Systems Inc.
12872 Valley View Avenue,
Suite 11B
Garden Grove, CA 92645
(714) 895-3381

Program Name: MDCSTAT 2.0
Hardware system: Any CP/M system
Minimum memory size: 48K recommended (can run in less memory)
Language: MBasic
Description: A fully interactive menu-driven statistics package. Includes a sophisticated data editor, and variables can be transformed by a dozen functions: e.g., multiply 2 columns to obtain new variable. Help can be obtained from manual on disk. Any function can be run with single key stroke. For example, "A" will execute an analysis of variance (1 and 2 way), "P" a paired t-test, "M" for multiple regression with ANOVA and analysis of residues, etc. F and t values are computed when needed. Treats missing values. Includes least square error function fitting to over 100 different functions. All output is formatted and can be routed to any system device. Plots histograms or x versus y on any standard plotter.
Release: 1978, Version 1.0; 1982, Version 2.0
Price: $49.95
Included with price: Floppy disk, with documentation on disk
Where to purchase it:
MDC
P.O. Box 115
Novato, CA 94968
(415) 883-9255

Program name: Fancy Font®
Hardware system: Can run on any CP/M system
Minimum memory size: 48K
Language: BDS C
Description: SoftCraft's *Fancy Font* personal typesetting system allows the user to fully exploit the superb capabilities of the low-cost Epson printer. The Epson is capable of printing with a resolution of 25,920 dots per square inch. Until now, most of that resolution was lost during printing. The *Fancy Font* typesetting system remedies this situation.

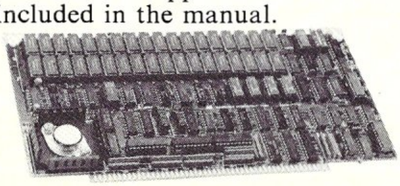
The *Fancy Font* system provides font sets in a large variety of styles sizes and faces, with sizes from 8 points to 21 points; styles including Roman, Sans Serif, Script and Old English; bold, italic, and regular faces. Parameters such as page size, tabs, margins, fonts, line spacing, headers, and footers can be specified when the document is printed.
Release: August 1982
Price: $180
Where to purchase it:
SoftCraft
8726 S. Sepulveda Blvd.
Suite 1641
Los Angeles, CA 90045
(213) 641-3822

Program name: PRIORITIES
Hardware system: 8080, Z80, CP/M, MP/M
Minimum memory size: 64K
Language: CB80
Description: PRIORITIES is a productivity-increasing tool for professionals, managers, and their staffs. Users receive a daily report organized for a workload mix of appointments and things to do (prioritized tasks). The daily report is divided into three sections: Appointments for day, tasks for day, and high priority tasks coming up. Tasks are kept in daily report until user marks them complete or reschedules them for a later date. Length of tasks coming up section can be regulated by increasing or decreasing days of warning or priority level.
Release: May 1982
Price: $99.50.
Included with price: Software, manual, diskbank
Where to purchase it:
BICS
P.O. Box 777
Pahala, HI 96777
(808) 928-8578

New Products

Multiuser memory board

Macrotech International Corp. has developed a 256-Kbyte dynamic memory board for the S-100 bus with addressing capabilities for both 8- and 16-bit systems; 24-bit addressing is supported for extended addressing systems. A mapping option, for use in systems with standard (16-bit) addressing capability, allows each 4K block of the 16-bit (64K) logical addresses to be translated to any 4K block of the 256K on-board physical memory. Bank switching is accomplished by reloading the appropriate map registers as required. A complete installation guide for MP/M II BMKXIOS and CP/M 2.2 Virtual Disk applications are included in the manual.



The model SS256 operates at 6 MHz with Z80/8085 processors, and at 8 MHz with most 16-bit processors without wait states. A faster version is available on request.

The board is available at $1379 list from: Macrotech International Corp., 22133 Cohasset St., Canoga Park, CA 91303; (213) 887-5737.

S-100 board with 32 DAC channels

Compatible with IEEE-696 microcomputer systems, the model SB-32-DA board has 32 12-bit digital-to-analog output channels. This extreme density is accomplished by multiplexing a single 12-bit converter.

In addition, each channel has its own 16-segment waveform generator. By writing to the SB-32's RAM, which is shared with on-board circuitry, the master processor can set 16 points on a waveform and 16 associated timing components. There are provisions for repeating waveforms, generating an interrupt upon completion of a waveform, and disabling any waveform generator so a channel may perform like a conventional memory-mapped DAC.

Besides the on-board RAM, the master may also directly access the board's time-of-day counter, three 16-bit counters, two time-of-day alarm comparators, three 8-bit ports, and eight frequency counter channels.

Five switchable unipolar and bipolar ranges are available. Each output can drive a 1K-ohm load and has 0.02% absolute accuracy.

Price: $825; Digital MultiMedia Control, 92972 River Road, Dept. NP, Junction City, OR 97448.

Sanyo desktop computer

Sanyo Business Systems Corp. has introduced a desktop computer model, MBC 1000. It utilizes a Z80A, 64K of RAM, CP/M, Sanyo Basic, diagnostics and utilities.

The unit features a green phosphor high impact non-glare CRT display, business graphics capability, a detachable keyboard with number pad, cursor control keys and five programmable function keys. The single 5¼″ double-sided double-density disk drive will have the add-on capability of three additional disk drives for a total 1.2M-byte disk storage capability.

A substantial library of business software has been developed in the United States to Sanyo's specifications. Sanyo will also offer word processing, electronic spreadsheets and related software to its distributor and dealer network.

In keeping with Sanyo's reputation for quality and service, Sanyo is extending the normal 90-day guarantee on all components other than disk drives to one year.

List price: $1,995. Sanyo Business Systems Corp., 51 Joseph St., Moonachie, NJ 07074; (201) 440-9300. Telex (WU): 642622.

Hard disk systems for S-100 and multibus

Winchester 8″ hard disk subsystems, which have a capacity of 10 to 120 megabytes and use 3M drives, are now being offered by Computex Microcomputer Systems. Prices range from $2,995 for a single 10-megabyte drive in a dual-drive cabinet with dual power supply and cable, making it suitable for expansion, to $9,496 for the two-drive 120Mbyte system. Other systems are available in 20-, 30-, 40- and 60-Mbyte configurations.

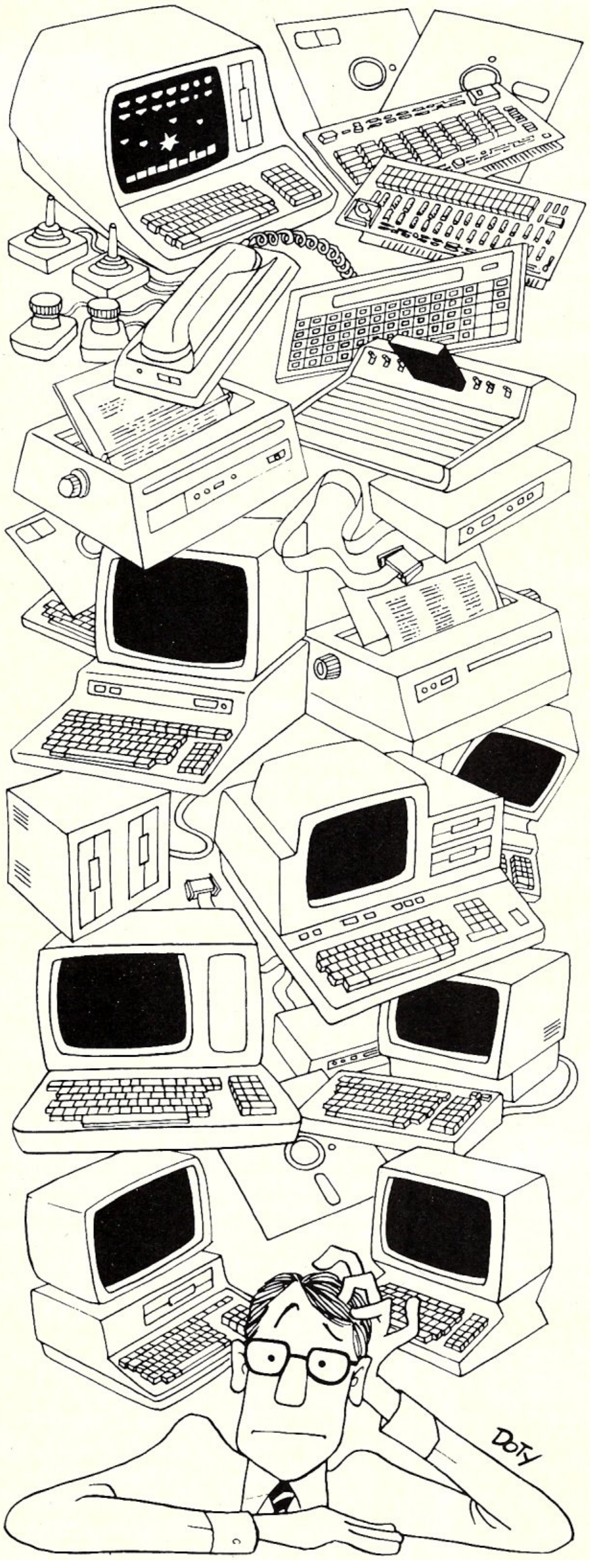
The S-100 controller board (from XCOMP) features multibank writable control memory for ultimate flexibility, with a disc data rate up to 10 MHz, an 8-bit bus transfer, 256-byte buffer, and is capable of 6 MHz IEEE-696 operation on the S-100 bus. In addition, the controller can be used in programmed I/O or interrupt mode.

The price of the controller is $898. Computex Microcomputer Systems, 5710 Drexel Ave., Chicago, IL 60637; (312) 648-3183.

Converter provides CP/M to IBM communications

Alphamatrix Inc. has introduced a new, cost-effective protocol converter for attachment to a CP/M-based computer. Designated C-BAX, the converter enables direct access to

NEED HELP BUYING A COMPUTER OR PERIPHERAL?



If you're shopping for a personal computer, for peripherals, for games—or practically anything that uses microcomputer technology—you need help. There are just too many products on the market for any one person to sort out.

The *Creative Computing 1983 Buyer's Guide to Personal Computers, Peripherals and Electronic Games* contains all the help you'll need to shop wisely. Its honest evaluations are written by experts—the experts of *Creative Computing.*

In the *Buyer's Guide* you'll find:

- How to select the computer for you
- Evaluations of specific personal computers
- Tips on buying peripherals of all kinds
- Reviews of peripherals
- Music synthesis equipment
- Comprehensive evaluation of joysticks, paddles and game port extenders
- Video game systems and software
- Roundup of electronic toys and games
- Electronic and computerized learning aids
- Video products for computer users
- Selected microprocessor-based consumer electronics products.

As you can see, the *Buyer's Guide* is an incredibly comprehensive directory to the latest in computer technology. Use the coupon below to order your copy today!

ORDER YOUR COPY TODAY!

Creative Computing
Buyer's Guide to Personal Computers
P.O. Box 640, Holmes, PA 19043

YES! Please send me the *Creative Computing 1983 Buyer's Guide to Personal Computers, Peripherals and Electronic Games.* I enclose $5.00 ($3.95* plus $1.05 postage and handling). Price outside U.S.A. $6.00.

Mr./Mrs./Ms.____________________
(please print full name)

Address____________________Apt.____

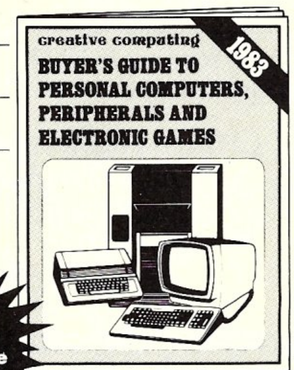
City____________________

State____________________

Zip____________________

*NJ residents add 5% sales tax.

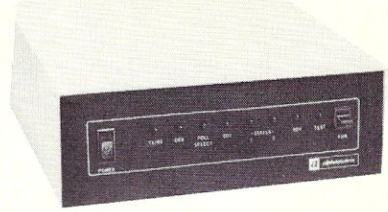
Please make check payable to *Creative Computing Buyer's Guide.*



GET THE GUIDE!

JUST $3.95 plus postage and handling

New Products
continued . . .



IBM and other large host data communications systems by allowing the CP/M to operate as if it were one of the host's own remote devices.

The C-BAX unit is modem-sized and attaches to a CP/M computer's serial I/O port. It requires no additional software.

The standard C-BAX converter for 2780/3780/3741 BISYNC communications costs $1,900 in single-unit quantity. Alphamatrix Inc., 1021 Millcreek Dr., Feasterville, PA 19047; (215) 355-3297.

New protocol converter

The Universal Protocol Converter, UPC-80, a new product from Equinox Data Systems, is intended to operate with a universal set of input and output data protocols (any mixture of serial or parallel formats). The UPC-80 will also perform string/code conversions, i.e., ASCII to EBCDIC. It has 6 bidirectional ports (3 serial and 3 parallel). Of the 3 serial ports, 2 will support synchronous byte protocols such as 2780 or 3270. They will also support asynchronous communications. The third serial port is synchronous and supports either SDLC or HDLC communications. All 3 serial ports are RS-232C compatible, with baud rates up to 19.2 KB. The three parallel ports are TTL compatible.

The UPC-80 consists of a single board containing a 4MHz Z80, 16K of dynamic RAM (operational storage), 8K of EPROM (firmware), and 1K of nonvolatile RAM (holds alterable system parameters).

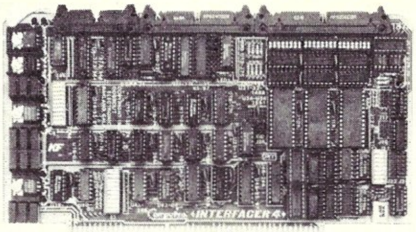
The board is populated only with the functions that a customer requests. The basic board includes firmware and hardware for one input and one output port. An example might be 9 parallel lines in and 2780 BISYNC out. The cost of firmware integration is variable and must be quoted from factory; however, charges for a typical single input and output program will average $650, making the average total cost under $1,500.

Equinox Data Systems, 517 Newman Springs Rd., Lincroft, NJ 07738; (201) 530-0505.

S-100 I/O board for up to 32 users

A serial/parallel interface board that allows as many as 32 contiguous users at eight port locations is now available from CompuPro. Conforming to all S-100/IEEE-696 timing specifications, the new Interfacer 4 incorporates an asynchronous serial interface and two synchronous/asynchronous high-speed channels. It also includes a Centronics-style parallel interface and a universal parallel port that can be used for custom interfacing.

A user selection port permits the cascading of up to eight Interfacer 4 boards at the same port address, thereby increasing the efficiency of the software, especially in multiuser environments. Interfacer 4 also features selectable 0, 1, 2 or 3



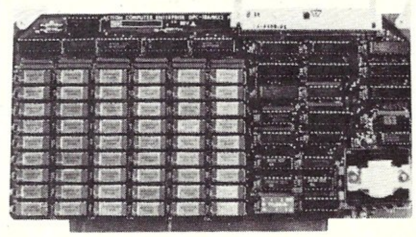
wait states for system operation at more than 10 MHz, and switch-selectable port addressing to any eight-port block. In addition, the board's interrupt structure offers a full masking and a flexible strapping capability to facilitate multiuser operation.

Price: $350. CompuPro Systems, Oakland Airport, CA 94614; (415) 562-0638.

Memory expansion for 16-bit processor

The DPC-186/MX board is a memory expansion board for Action Computer Enterprise's 16-bit 8086/8087-based DPC-186 user processor for Discovery microcomputer systems. Up to three MX boards can be connected to the DPC-186 to configure a full 1-Mb processor. The DPC-186/MX is available in three options: 128K, 256K, and 384K. Since the DPC-186 already includes 128K on board, the two-board set consisting of the DPC-186 and only one fully populated MX board will provide the user

New Products
continued . . .

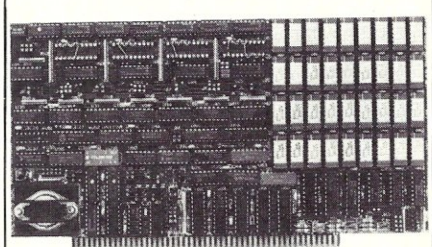


with a half-megabyte of memory for the 8086. The 8087 Numeric Data Processor on the DPC-186 board adds extremely high-performance computational power, including all floating-point and transcendental functions with 80-bit precision.

Price: 128K, $1,095; 256K, $1,595; 384K, $1,995. Action Computer Enterprise, Inc., 55 W. Del Mar Blvd., Pasadena, CA 91105; (213) 793-2440

256K S-100 memory board

The 256KMB-100 from Intercontinental Micro Systems is an S-100 bus, 256K memory board featuring bank select in any combination of 16K banks. It has phantom deselection, deselect on 4K boundaries from 4K to 64K, and operates at 4MHz with no wait states. Parity errors are detected through interrupts. A complete one-year



warranty for parts and labor is also provided. Price: 256K, $1,390; 128K, $1,125.

Intercontinental Micro Systems, 1733 So. Douglass Rd., Suite E, Anaheim, CA 92806; (714) 978-9758.

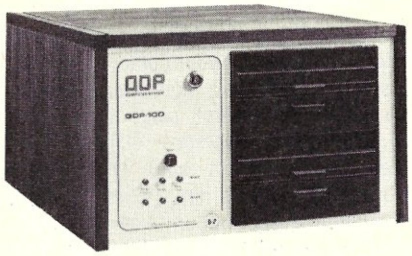
S-100 computer system

QDP-100 is a Z80A system with the following features:
• 64KB (4MHz) dynamic RAM, bank selectable in 16KB segments • Up to 8KB of

ROM (2716) • 2.4MB storage on 2 double-sided, double-density 8″ Qume (Qumetrak 842) floppy drives. Supports single-density read/write/format option (IBM-3740). Automatic density and sector size selection • Cache memory system available with throughput several times faster than competitive systems, and reduced disk wear • 2 serial/2 parallel ports (one Centronics-compatible), with up to 6 serial and 4 parallel ports for multiuser applications • RS-232-C interface. Full port interrupt/handshaking capabilities • 30-character serial port input buffer memory • Full IEEE-696 compatibility • Modular construction for ease of expansion and servicing • 6-slot S-100 motherboard with three available expansion slots in a typical system • Programmable drive shut-off timer to protect media and drives • Front panel status indicators for each disk drive.

Fully defined MENU-guided configuration program allows operating system parameters to be easily changed through keyboard input.

The comprehensive HELP system shortens learning time



and guides operators through their tasks with options, definitions, examples and in-depth system operating information. Informative MENU capability offers complete on-line menu-style display listings of available programs and options that can be selected and executed.

MP/M version 2.1 is provided for multiuser systems and is easily added to existing systems. In a typical application, up to four concurrent operating tasks are performed. Password security is also available with MP/M.

CP/M version 2.2 and CBasic are standard. Other leading languages also available. The on-line Error Diagnostic routine explains errors fully. Trap Error feature allows operator to immediately retry error step or ignore a piece of faulty data and continue with good data. Auto-load command assignable to bring a designated program immediately on-line.

Price: $4,695 with a one-year warranty. QDP, 10330 Brecksville Rd., Brecksville, OH, 44141; (216) 526-0838.

New Products
continued . . .

S-100 interface for non-bus systems

Kramer Systems International, Inc. (KSI) has available an interface that allows any Superbrain, Compustar, Televideo, or any Z80A-based computer to directly control devices connected to an S-100 bus. The KSI S-100 Interface is compatible with IEEE S-100 Bus Standard Specification 696.

The KSI S-100 Interface generates the majority of S-100 signals; capabilities include: Superbrain/Televideo as permanent bus master; 8-bit path between Superbrain/Televideo and S-100 Bus; 64Kbyte S-100 memory space (standard addressing) plus 64Kbyte S-100 I/O space (extended addressing), in addition to Superbrain/Televideo's internal address space; functional support of interrupts, DMA, and temporary bus masters; all read/write data, address, status, and control bus signals generated per IEEE-696. Access to S-100 bus is accomplished by simply reading or writing data to one of six Superbrain/Televideo I/O ports.

The KSI S-100 Interface consists of two printed circuit boards, a small "top-hat" board that plugs into the primary Z80A socket within the Superbrain/Televideo, and a second board that plugs directly into an S-100 card cage. A single-ribbon cable interconnects the two circuit boards.

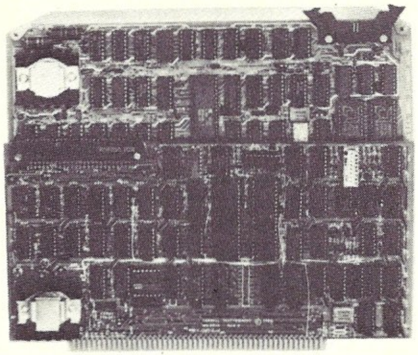
Price: $600. Kramer Systems International, Inc., 8403 Dixon Ave, Silver Spring, MD 20910; (301) 585-7480.

CPU double-density package

A ready-to-go CPU double-density board package is now offered by Tarbell Electronics that includes their Z80A S-100 CPU I/O board, double-density floppy disk interface, Digital Research CP/M, and Tarbell's new disk cache memory system. Tarbell's CPU board features memory-management hardware that allows dynamic mapping of up to 1Mbyte of memory in 4K blocks. The on-board table memory may be loaded by the programmer or system software with I/O commands.

The double-density floppy disk interface includes true Direct Memory Access (DMA) in single- or double-density modes which make possible efficient multiuser and multitasking operations. The interface features on-board phase-locked loop and write-precompensation circuits that provide highly reliable operation with a variety of drives. All 64Kbytes of address space are available, since the board is not memory mapped. Tarbell's cache BIOS system keeps the most recently used disk sectors in memory for easy access; increases overall speed of operations 2 to 4 times for CP/M and 2 to 3 times for MP/M.

Price is $945. Tarbell Electronics, 950 Dovlen Pl., Suite B, Carson, CA 90746; (213) 538-4251.



Desktop multiuser system

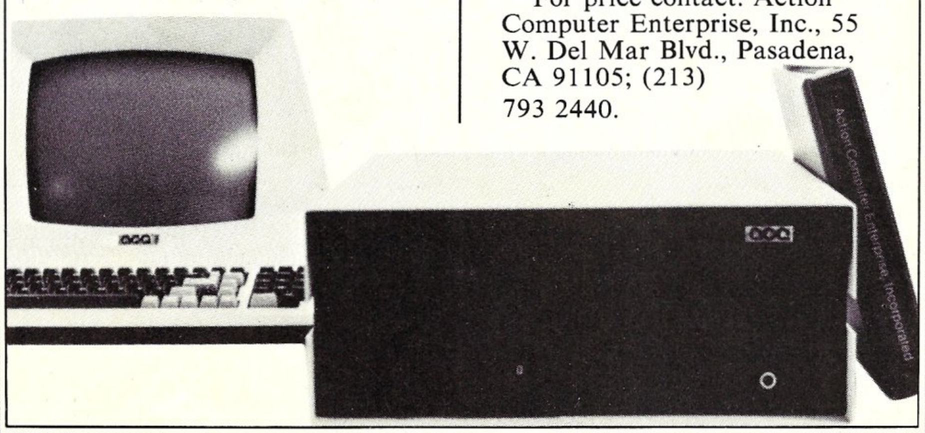
Housed in a single 17″ x 21″ x 8½″ cabinet, the Desktop Discovery from Action Computer Enterprise, Inc., serves up to six user terminals and two printers simultaneously. A built-in Winchester hard disk drive is available with up to 20 Mb of mass storage capacity. Backup is provided by a 5¼″ mini-floppy disk capable of storing 640Kbytes.

This Desktop Discovery can be configured as a single-user computer powered by a Z80A CPU at 4 MHz with 64Kbytes of user RAM memory running the CP/M-80 operating system. Additional user processors can be added in any mix of Z80A-based 8-bit processors, or 8086/8087-based 16-bit processors with 128Kbytes of on-board RAM apiece. Sixteen-bit user processors run CP/M-86 and can be expanded to 1 Mb of RAM each, with some restrictions on the number of users when expansion memory is added.

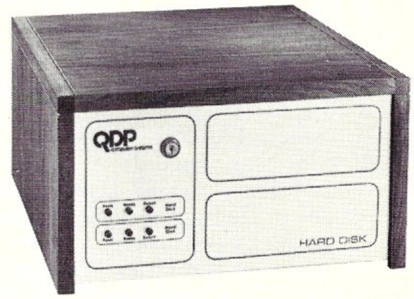
Multiuser configurations include Action's DPC/OS multiuser operating system. Available application software includes any CP/M-80 or CP/M-86 compatible programs.

The Desktop Discovery fills in the low end of Action Computer's product line. The original Discovery, in production since 1979, handles 16 user terminals and can be supplied with 8″, 10½″, or 14″ Winchester disks with up to 120 Mb of mass storage. Streaming tape, cartridge disks, or 8″ floppy disks provide data backup on the full size Discovery.

For price contact: Action Computer Enterprise, Inc., 55 W. Del Mar Blvd., Pasadena, CA 91105; (213) 793 2440.



S-100 disk storage units



The QDP HD-10 and HD-15 storage expansion units are self-contained computer add-ons featuring a 5¼″ micro-Winchester hard disk drive with controller and power supply. These units provide a formatted capacity of 10- or 15Mbytes with a transfer rate of 5Mbits per second to the S-100 bus and a track-to-track access time of 2 msec. Features are: Front panel drive status indicators and primary power turn-key • Heavy duty, dual voltage (110/220 VAC, 60/50 Hz) power source, filtered from noise and spikes for industrial applications. Uninterruptible power source also available • Modular construction for ease of servicing • Sturdy lightweight contruction with quiet, filtered cooling system for increased equipment life • QDP pride in product quality and reliability. Typical drive usage reliability is: MTBF 8000 continuous operational hours • The HD-10 is priced at $3,995 and the HD-15 is $4,995, both with interface cable, a one-year warranty and complete documentation.

QDP Marketing Director, 10330 Brecksville Rd., Brecksville, OH 44141; (216) 526-0838.

Complete 10-megabyte microcomputer system

A newly designed 10Mbyte microcomputer system has been released by the IMSAI Division of the Fischer-Freitas Corporation.

It includes: 10Mbyte hard disk, one 5¼″ single-sided, dual-density floppy disk backup, 8-bit microprocessor; memory-mapped video display board, disk controller, 10-slot S-100 motherboard; 28-amp power supply, 12″ video monitor, 62-key ASCII keyboard, 132-column dot matrix printer, CP/M and 64K RAM. Options include 5- or 18Mbyte hard disk, 16-bit microprocessor, 256K RAM, 86-key ASCII extended keyboard, and single or double dual-density 8″ floppy disk drives.

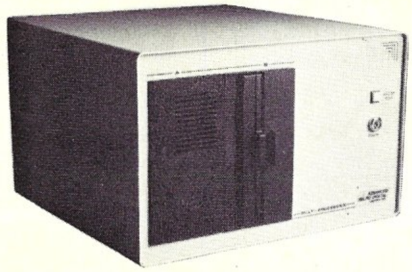
Price: $5,995. IMSAI Division, Fisher-Freitas Corp., 910 81st Ave., Oakland, CA 94621; (415) 635-7615.

New Products continued . . .

New multiuser system uses master/slave processors

Advanced Microdigital Corp. has introduced "Super-System," an S-100-based computer using a unique master/slave concept that provides each user with his own dedicated Z80 CPU. An unlimited number of users can operate on the Super-System without noticeably eroding performance levels.

The basic system includes a power supply, chassis, 8-slot

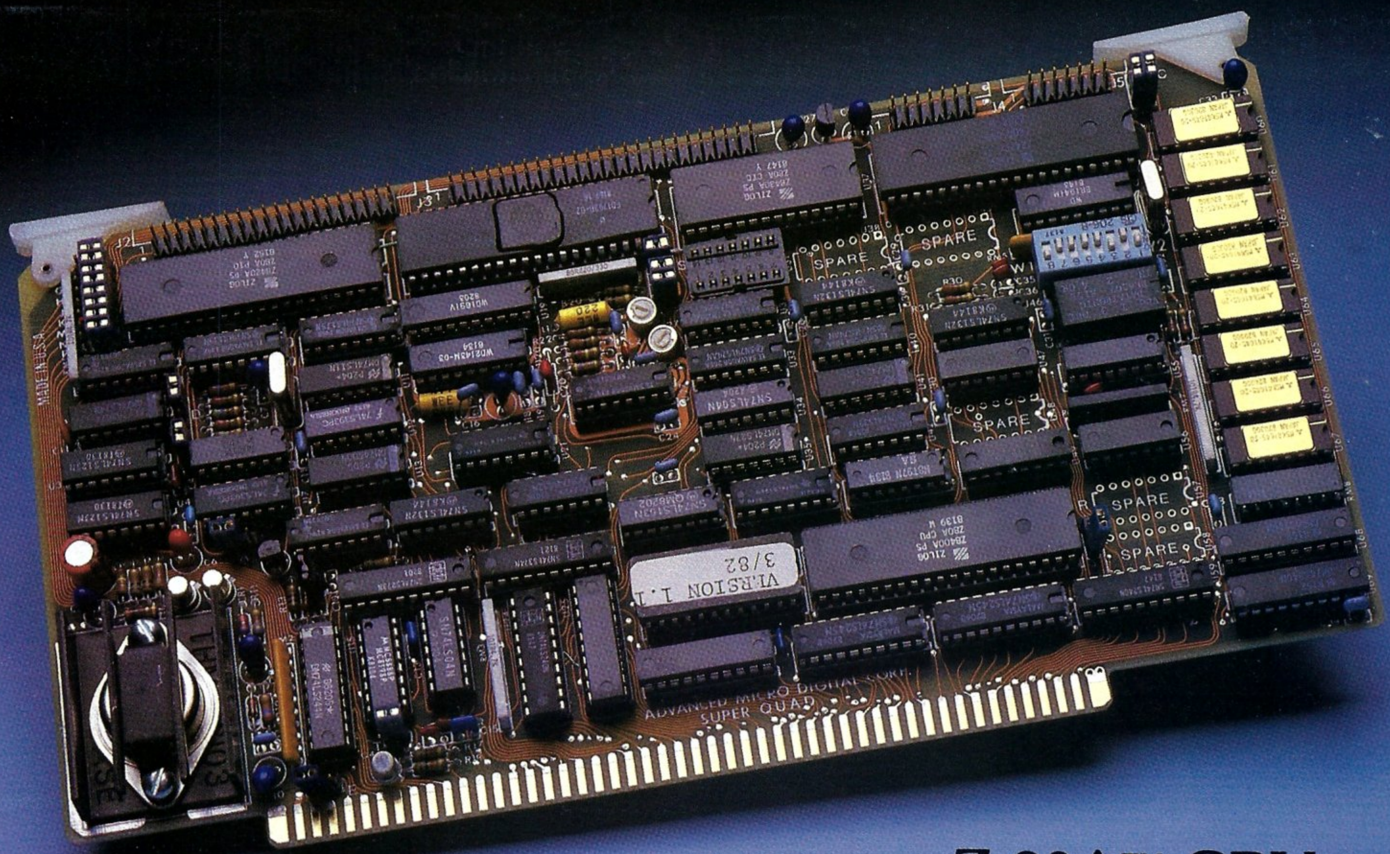


motherboard, and two Shugart 801R single-sided drives. Delivered with the CP/M, it lists for $2,675. By adding Super-Slave processors to the computer's motherboard, the system grows into a multiuser, multiprocessing configuration with no limitations on its eventual size. The slave boards provide each user with his own dedicated CPU, four serial and two parallel interface ports, 2K (expandable to 4K) of EPROM and 64K (expandable to 128K) of bank-switchable RAM—all for an additional $650 cost. Advanced Microdigital Corp., 12700 B Knott Ave., Garden Grove, CA 92641; (714) 891-4004.

Microsystems Mart

ADVERTISERS INDEX

The World's First



Z-80A™ CPU, Floppy Disk Controller, 64K of Memory, Serial & Parallel I/O Ports . . . all on a SINGLE S-100 BOARD!

Don't Buy Another One of Those 3 Board Sets Till You See What One Can Do!

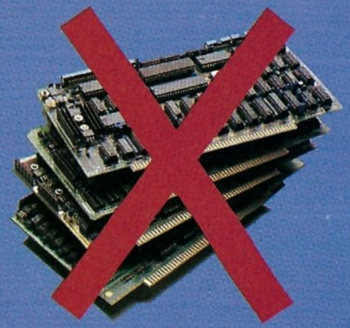
Advanced Digital has been producing the SUPER QUAD for some time now and it's truly one of a kind. Just plug this board into any S-100 mother board and hook-up your disk drives to it. It flies. Runs with CP/M, MP/M, and turbo-DOS. You can also plug in additional boards, I/O, hard disk controllers, etc. SUPER QUAD is a BUS master. The cost of this board is one third to one half of what you have been paying for the three board set. Just take a look at these features:

®Registered Trademark of Digital Research Corp.
™ Registered Trademark of Software 2000 Inc.

- IEEE S-100 Standard
- Z-80A CPU
- 64K of Bank Select Memory as well as extended addressing
- Double density floppy disk controller. Both 8 or 5¼" Disk Drives
- 2 serial & 2 parallel I/O ports (RS-232 and intelligent hard disk interface).
- 2K or 4K of monitor EPROM
- Runs with CP/M®, MP/M® and turbo-DOS™
- One year warranty.
- Free copy of bios disk.

Also introducing SUPER/SLAVE to run multi-processing operating systems such as turbo-DOS™. With 128/64K of memory, serial & parallel I/O, you can plug multiple of the slaves with the master and each user will have its own CPU and memory local.

CIRCLE 23 ON READER SERVICE CARD



ADVANCED DIGITAL CORPORATION

Now it's time to replace or throw away the old S-100 boards and plug the SUPER QUAD in. Save space, money and power.

Retail price is $875.

For more information write or call: Sales Dept.

12700-B Knott Street • Garden Grove, California 92641 • (714) 891-4004 TELEX 678401 tab irin

CompuPro Spoken Here ...8 and / or 16 Bits

CompuPro speaks your language - 8 bit, 16 bit, or 8 **and** 16 bit - while delivering exceptional performance and reliability. As **Interface Age** said about our ground-breaking dual **CPU 8085/88** processor, "The (8 bit) 8085 was more than a third faster than any 8 bit micro we have tested to date...the (16 bit) 8088 (was) almost twice as quick as the identically-engined IBM Personal Computer utilizing a similar software package". The **Silicon Gulch Gazette** described **CompuPro** machines as "Definitely reliable and potent...likely to be a major product for serious technologists who demand trustworthy hardware".

We couldn't have said it better. See what people are talking about at your nearest **CompuPro Authorized Systems Center.**

Byte Shop of Hayward
1122 B Street
Hayward, CA 94541
(415) 537-2983

Computer Center, Inc.
1514 University Avenue
Berkeley, CA 94703
(415) 845-6366

Digital Deli
80 W. El Camino Real
Mountain View, CA 94040
(415) 961-2670

G & G Engineering
1922 Republic Avenue
San Leandro, CA 94577
(415) 895-0798

G & G Engineering
230 California Street
Suite 207
San Francisco, CA 94104
(415) 391-4570

JR Systems, Inc.
8617 Bunnell Drive
Potomac, MD 20854
(301) 299-4522

Key Microsystems
978 Tiogue Avenue
Coventry, RI 02816
(401) 828-7270

Key Microsystems
822 Boylston St.
Chestnut Hill, MA 02167
(617) 738-7305

Logic Systems
4800 Manzanita Avenue #21
Carmichael, CA 95608
(916) 971-3133

Matrix Computer Systems
720 Mendocino Avenue
Santa Rosa, CA 95401
(707) 542-0571

Micro Computer Technology
1549 W. Brandon Boulevard
Brandon, FL 33511
(813) 685-7659

Priority One Electronics
9161 Deering Avenue
Chatsworth, CA 91311
(800) 423-5922, (213) 709-5464

S-100 Inc.
14425 N. 79th St. #B
Scottsdale, AZ 85260
(800) 528-3138, (602) 991-7870

CompuPro SYSTEMS

CompuPro division, Godbout Electronics,
Box 2355 Oakland Airport, CA 94614

CIRCLE 13 ON READER SERVICE CARD