

OPERATION MANUAL

USER'S GUIDE

SOFTWARE SPECIFICATION GUIDE

IN-CIRCUIT DEBUGGER **ICD-278** FOR Z80

ZAX
Zax Corporation

Preliminary

ICD 278

FOR

Z80

OPERATION MANUAL

Copyright© 1984, ZAX Corporation. All Rights Reserved under International and Pan-American Copyright Conventions. No part of this publication may be reproduced in any manner whatsoever without written permission from ZAX Corporation.

Prepared by

ZAX Corporation
2572 White Road, Irvine, California 92714

Operation Manual: Part No. 20-100-06, Rev.B includes;

User's Guide (278/Z80)	: Part No. 20-100-XX
Software Specification Guide (278/Z80)	: Part No. 20-100-XX
Command Reference Guide	: Part No. 20-100-03

Printed: June 1984

LIST OF EFFECTIVE PAGES

THE TOTAL NUMBER OF PAGES IN THIS PUBLICATION IS 151, CONSISTING OF:

PAGE No.	ISSUE DATE/REVISION LEVEL
TITLE	JUNE 1984/REV.B
B	JUNE 1984/REV.B
I THROUGH VII	JUNE 1984/REV.B
1-1 THROUGH 1-6	JUNE 1984/REV.B
2-1 THROUGH 2-8	JUNE 1984/REV.B
3-1 THROUGH 3-40	JUNE 1984/REV.B
4-1 THROUGH 4-2	JUNE 1984/REV.B
4A-1 THROUGH 4A-4	JUNE 1984/REV.B
4B-1 THROUGH 4B-2	JUNE 1984/REV.B
4C-1 THROUGH 4C-16	JUNE 1984/REV.B
4D-1 THROUGH 4D-4	JUNE 1984/REV.B
4E-1 THROUGH 4E-12	JUNE 1984/REV.B
4F-1 THROUGH 4F-4	JUNE 1984/REV.B
4G-1 THROUGH 4G-8	JUNE 1984/REV.B
II	JUNE 1984/REV.B
5-1 THROUGH 5-12	JUNE 1984/REV.B
6-1 THROUGH 6-20	JUNE 1984/REV.B
A THROUGH B	JUNE 1984/REV.B

NOTE: THE PAGES AFFECTED BY CHANGES ARE NOTED BY * AND INDICATED BY ISSUE DATES AND REVISION LETTERS PRINTED NEXT TO THE PAGE NUMBER. THE ORIGINAL ISSUE OF THIS PUBLICATION IS REVISION B, JUNE 1984.

TABLE OF CONTENTS

PART I - USER'S GUIDE

Section		Page
I	INTRODUCTION and DESCRIPTION	
	Introduction	1-1
	Description	1-1
	ICD Major Components	1-2
	Specifications (General)	1-3
	Specifications (Emulation)	1-4
II	CONTROLS and COMPONENTS	
	Introduction	2-1
	ICD Control and Component	
	Description	2-1
	Top View	2-1
	Side View	2-1
	End View (A)	2-2
	Baud Rate Switch Settings	2-3
	End View (B)	2-4
	External Break Connector	2-5
	Event Trigger Connector	2-5
	CPU Plug Probe	2-6
	External Break Cable	2-7
	External Trigger Cable	2-7
III	SYSTEM CONFIGURATION/START-UP SEQUENCE and COMMAND GUIDE	
	Introduction	3-1
	System Configuration	3-2
	Start-up Sequence	3-3
	Command Guide	3-5
IV	TECHNICAL REFERENCES	
	Introduction	4-1
	A - Emulation Select Switch	4A-1
	B - Data Bus Emulation Connector	4B-1
	C - Serial Interface	4C-1
	S-791 SIO Module Description ...	4C-1
	Transmission Format Switch	4C-2
	Serial Interface	
	Specifications	4C-3
	RS-232C Interface (Terminal) ...	4C-4
	RS-232C Interface (Host/Aux) ...	4C-5
	Current Loop Interface	4C-7
	TTL Interface (Terminal)	4C-9
	TTL Interface (Host/Aux)	4C-10
	Control Signal Timing	4C-12

Section

Page

IV
(cont.)

D - ICD Program Memory	4D-1
User Memory	4D-2
Memory Mapping	4D-3
E - CPU Emulation	4E-1
ICD Program Memory Cycle	4E-2
Z80 Machine Cycles	4E-3
Clock Switch	4E-5
Reset Switch	4E-7
Monitor Switch (Interrupt)	4E-8
Bus Control	4E-9
REFRESH Signal	4E-10
Signal Timing	4E-11
F - Probes	
Event Trigger Probe	4F-1
Emulation Qualify Output Probe	4F-2
Map Control Probe	4F-3
G - Control Modules	
Introduction and Description ...	4G-1
S-730 Indicator Panel	4G-2
S-791 Serial Interface Output ..	4G-3
S-795 Realtime Trace Storage ...	4G-4
S-793 Central Processing Unit ..	4G-5
S-792 Emulation Memory Unit	4G-6
ICD Disassembly/Module Removal	4G-7

PART II - SOFTWARE SPECIFICATION GUIDE

Section	Page
-	INTRODUCTION and DESCRIPTION
	Introduction II
	Description II
V	SOFTWARE COMMANDS
	Introduction 5-1
	ZICE Keywords 5-2
	ZICE Format 5-3
	Load Command 5-5
	Save Command 5-6
	Verify Command 5-7
	Quit Command 5-8
	Z Subcommands
	Z Log Subcommand 5-8
	Z Batch Subcommand 5-8
	Z Comment 5-9
	Z Wait Subcommand 5-9
	Z Directory Subcommand 5-10
	Z Help Subcommand 5-10
	Z Repeat Subcommand 5-11
	Z Print Subcommand 5-11
VI	HOST COMPUTER SYSTEM COMMUNICATION
	Introduction 6-1
	Communication Terminology 6-2
	System Configuration 6-3
	Idle
	Idle Program 6-4
	Idle Program Flowchart 6-5
	Display Text
	Display Text Sequence 6-6
	Display Text Sequence Diagram 6-6
	Display Text Program 6-7
	Display Text Flowchart 6-7
	Command Input Request Program
	Command Input Request Sequence ... 6-8
	Command Input Request
	Sequence Diagram 6-8
	Command Request Program 6-9
	Command Request Flowchart 6-9
	Object File Load/Verify
	Object File Load/Verify
	Sequence 6-10
	Object File Load/Verify
	Sequence Cont. 6-11
	Object File Load/Verify
	Sequence Diagram 6-11

Section

Page

VI
(cont.)

Object File Save	
Object File Save Sequence	6-12
Object File Save Sequence Diagram	6-13
Z Command	
Z Command Sequence	6-14
Z Command Sequence Diagram	6-14
Z Command Program	6-15
Z Command Flowchart	6-15
Quit	
Quit Sequence	6-17
Quit Sequence Diagram	6-17
Quit Program	6-17
Quit Flowchart	6-17
Display Symbolic Text	
Display Symbolic Text Sequence ...	6-18
Display Symbolic Text Sequence Diagram	6-18
Display Symbolic Text Program	6-19
Display Symbolic Text Flowchart ..	6-19
Console Key Check	
Console Key Check Sequence	6-20
Console Key Check Sequence Diagram	6-20

LIST OF ILLUSTRATIONS

Figure	Title	Page
2-1	ICD Controls and Components	2-1
2-2	ICD Controls and Components, cont. ...	2-2
2-3	ICD Controls and Components, cont. ...	2-4
2-4	CPU Plug Probe	2-6
2-5	External Break Cable Diagram	2-7
2-6	Event Trigger Cable Diagram	2-7
4-A.1	Emulation Select Switch	4A-1
4-A.2	E. S. Switch Factory Settings	4A-1
4-A.3	Emulation Data Bus Block Diagram	4A-2
4-A.4	Emulator READ Signal Block Diagram ...	4A-3
4-A.5	Wait States	4A-4
4-B.1	DB. EMUL (Data Bus Emulation) Connector	4B-1
4-B.2	DB. EMUL Connector Configuration	4B-2
4-C.1	S-791 SIO Module	4C-1
4-C.2	Transmission Format Switch	4C-2
4-C.3	RS-232C Standard Jumper Settings	4C-6
4-C.4	CLI Standard Jumper Settings	4C-8
4-C.5	TTL Standard Jumper Settings	4C-11
4-C.6	BUSY and DTR Input Signal Diagram	4C-12
4-C.7	BUSYOUT and DSR Output Signal Diagram	4C-13
4-C.8	RSTP Output Signal Diagram	4C-13
4-C.9	S-771 SIO Interface Circuit Diagram (Terminal)	4C-14
4-C-10	S-771 SIO Interface Circuit Diagram (Host/Aux)	4C-15
4-D.1	ICD Program Memory Diagram	4D-1
4-D.2	User Memory Timing Diagram	4D-2
4-D.3	Mapping Configuration	4D-3
4-E.1	Emulator/Target System Interface	4E-1
4-E.2	Clock Configuration	4E-5
4-E.3	Setting of Internal Clock	4E-6
4-E.4	Reset Configuration	4E-7
4-E.5	Interrupt Signal Configuration	4E-8
4-E.6	BUSRQ Signal Configuration	4E-9
4-E.7	REFRESH Signal Configuration	4E-10
4-F.1	Event Trigger Output Timing Diagram ..	4F-1
4-F.2	Emulation Qualify Output Timing Diagram	4F-2
4-F.3	Map Control Timing Configuration	4F-3

Figure	Title	Page
6-1	System Configuration	6-3
6-2	Idle Program Flowchart	6-5
6-3	Display Text Sequence Diagram	6-6
6-4	Display Text Flowchart	6-7
6-5	Command Input Request Sequence Diagram	6-8
6-6	Command Input Request Flowchart	6-9
6-7	Object File Load/Verify Sequence Diagram	6-11
6-9	Object File Save Sequence Diagram	6-13
6-11	Z Command Sequence Diagram	6-14
6-12	Z Command Flowchart	6-15
6-13	Quit Sequence Diagram	6-17
6-14	Quit Flowchart	6-17
6-15	Display Symbolic Text Sequence Diagram	6-18
6-16	Display Symbolic Text Flowchart	6-19
6-17	Console Key Check Sequence Diagram ...	6-20

LIST OF TABLES

Table	Title	Page
2-1	ICD Baud Rates	2-3
2-2	External Break Connector	2-5
2-3	Event Trigger Connector	2-5
3-1	System Configuration	3-2
3-2	Start-Up Sequence	3-3
4-C.1	RS232C Interface Pin Designation (Terminal)	4C-4
4-C.2	RS232C Interface Pin Designation (Host/Aux)	4C-5
4-C.3	Current Loop Interface Pin Designation (Term/Host/Aux)	4C-7
4-C.4	TTL Interface Pin Designation (Terminal)	4C-9
4-C.5	TTL Interface Pin Designation (Host/Aux)	4C-10
4-C.1	Emulator Bus Connector - Pin Assignment	4C-5
4-E.1	ICD-278 Signal Timing Diagram	4E-11

SECTION I

INTRODUCTION and DESCRIPTION

INTRODUCTION

This Operation Manual contains information concerning the ZAX ICD-278 for Z80 IN-CIRCUIT EMULATOR. Revision and update information is contained in supplements issued periodically to Operation Manual recipients.

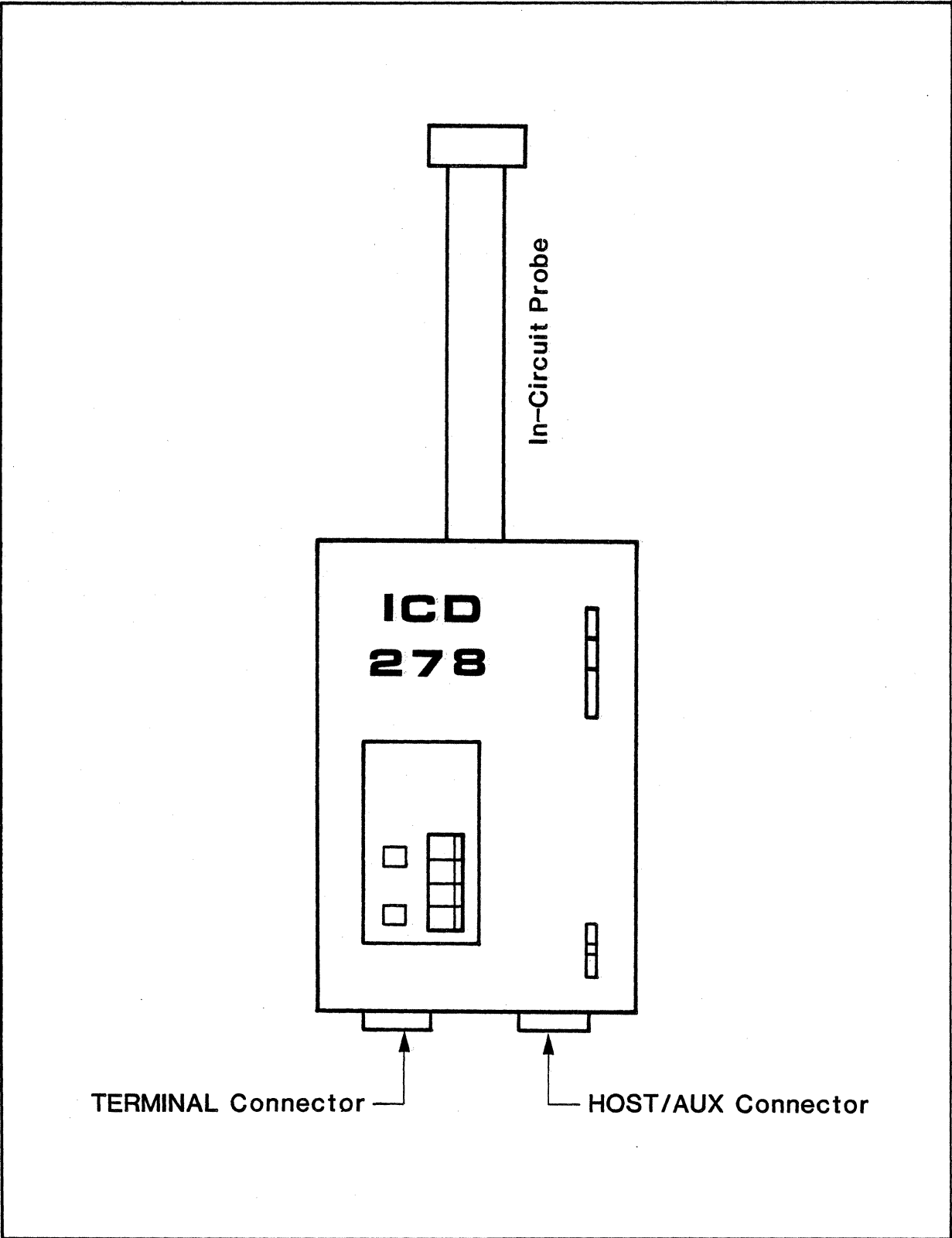
Part I of this manual acts as a USER'S GUIDE and describes the ICD-278, including controls, system configuration, and start-up sequence. A system command guide with general examples is included in Part I. This guide illustrates and explains the basic command syntax parameters and functions of the ICD commands. A complete listing of Technical References may also be found in Part I. ICD communication with a Host Computer system is contained in Part II; SOFTWARE SPECIFICATION GUIDE, and describes the proper connection, communication protocol and program construction.

Always refer to the specific information you are concerned with, as general knowledge of certain operations and procedures may not prove fully satisfactory in obtaining the desired results; moreover, the design engineer is highly encouraged to become familiar with the entire contents of this manual in order to thoroughly realize the ICD-278's capabilities.

DESCRIPTION

The ZAX ICD-278 IN-CIRCUIT EMULATOR functions as a stand-alone device for developing and maintaining hardware and software of Z80B microprocessors. The ICD-278 comes standard with 64K Bytes of static memory for mapping, displays in HEX/ASCII or disassembled Intel code with a 2K by 40 real time trace buffer and 29 major software commands.

For field applications, the ICD-278 can be connected to any communications device that has either RS232C or 20 mA current loop input.



ICD Major Components

SPECIFICATIONS (General)

Dimensions	:300mm (11.8in) wide :210mm (8.2in) deep : 80mm (3.2in) high
Probe Length	:510mm (20in) long
Weight	:3.3kg (7.3lb)
Power Requirements	:115VAC/230VAC;50/60Hz
Operating Temperature	:0°C to 45°C
Storage Temperature	:-10°C to 55°C
Humidity	:30% to 85%, relative (non-condensing)
Processors	:Z80 (to 6MHz) :i8085 (6MHz)
Memory Size	:64K-Bytes static RAM
Memory Mapping	:Full 64K byte
Real Time Trace Buffer	:2K x 32 bits
System Commands	:29
Usable I/O Ports	:256
Breakpoints	:3(hardware), 8(software) 1(external probe)
Communication Ports	:2 RS232C/20ma current loop/TTL
Baud Rates	:14 usable - to 19,200bps (factory set at 9,600bps)
Operating Temperature	:0°C to 45°C
Storage Temperature	:-10°C to 55°C
Humidity	:30% to 85%

SPECIFICATIONS (Emulation)

Applied CPU

Z80 : 500K, 2.5 MHz
Z80A : 500K, 4 MHz
Z80B : 500K, 6 MHz

Memory Area

Program memory :The entire area of the program memory (64K-bytes) is open. This memory is composed of high-speed static RAM.

User memory :The entire area of 64K-byte memory space is open to the target system.

Mapping :Both the program and user memory can be mapped in 1K-byte units. Four types of mapping modes are available;

User Memory
Emulation Read/Write Memory
Emulation Read Only Memory
Non Memory

I/O Port :All 256 ports are open.

Breakpoints :3 Hardware, 8 Software and External trigger break.

Hardware Break specification :A,B,C and Event trigger
:Specify enable/disable of A,B,C and Event breakpoints.

A,B,C Breaks :Address 16 bits, BHE. Each bit may be specified 0,1 or "don't care."

Status: OP code fetch
Memory access
Memory read
Memory write
I/O access
I/O read
I/O write
Execution of instruction

Arm: Break occurs only at the breakpoint after an event trigger is generated.

Event Trigger
break :Address 16 bits, BHE. Each bit may be specified 0,1 or "don't care."

Status:
 OP code fetch Memory access
 Memory read Memory write
 I/O access I/O read
 I/O write Instruction execution

Data: 8 bits. Each bit may be specified 0,1 or "don't care."

External Trigger
break :Breakpoint; 1 channel - TTL level Functional specification; HIGH edge or LOW edge.

Software Break :8 points; 0 - 7
 User breakpoint :Any point may be specified by using the LDA, A instruction.

Specification :Specify enable/disable of all software points.

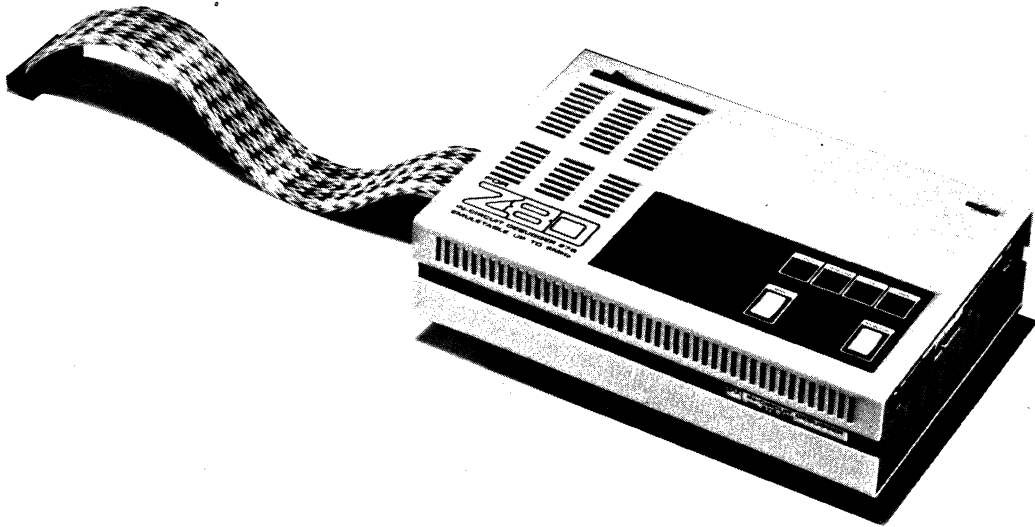
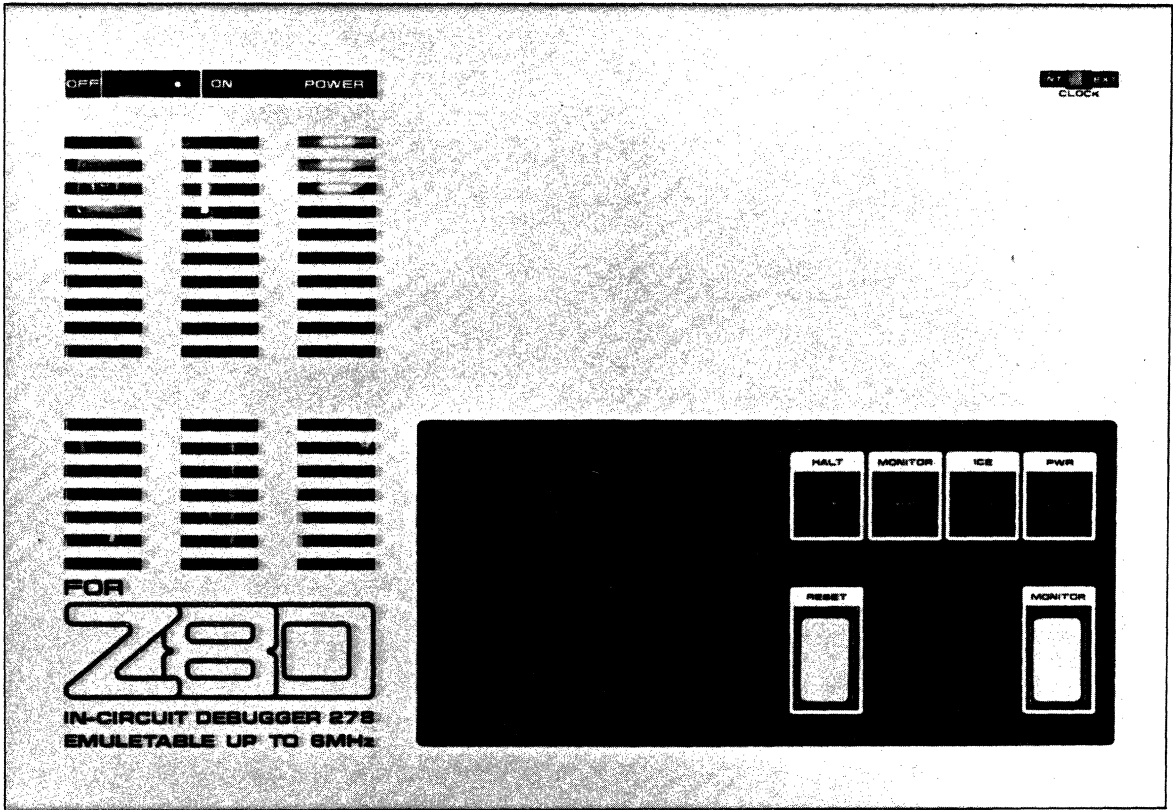
Function :A break is caused in the target system when the CPU reads 7FH as an OP code (which represents a LDA,A instruction). Its execution does not effect the registers or flags. Software breaks are suppressed when disabled.

Realtime trace :The addresses, data and status during emulation may be stored in storage memory in realtime.

Trace capacity :2K x 32 bits
 Fixed trace data :A0-15, D0-7, MREQ;IORQ, RD;WR, MI

Trigger function :End monitor
 Begin monitor
 End event
 Begin event
 Center event
 Multiple event

Realtime counter :Count bit width; 32 bit
 Count time; 716 sec, 6 MHz



SECTION II

CONTROLS and COMPONENTS

INTRODUCTION

This section introduces the various controls, components and functions of the ICD-278. The system controls are represented by four different views of the unit. The user may also wish to examine the photographs of the ICD-278 found at the beginning of Section I.

- 1 POWER switch. This switch is used to supply power to the ICD-278.
- 2 CLOCK select switch. This switch is used to select between the external target (EXT) and the internal clock (INT).
- 3 RESET switch. This momentary switch is used to reset the ICD-278 monitor. It is activated when the MONITOR lamp is on.
- 4 MONITOR break switch. This momentary switch is used to return control to the ICD monitor during emulation.
- 5 HALT lamp. This LED comes on when the ICD CPU has stopped after executing a HALT instruction or when BUSY ACKNOWLEDGE (BUSAK) is in progress.
- 6 MONITOR lamp. This LED comes on to indicate that control is currently in the ICD monitor. It is off during emulation.
- 7 ICE (In-circuit Enable) lamp. This LED comes on when the ICD is operating in the incircuit mode (I1 or I2) with the target system.
- 8 POWER lamp. This LED comes on when the ICD is on.
- 9 POWER select switch. This switch is used to select the proper power requirements for the ICD. Set the switch to 110/117V to run on a power supply of 110-120VAC or select 200/240V to run on a power supply of 200-240VAC. Set this switch BEFORE connecting the power plug.
- 10 Power connector. This connector is for the AC power plug.

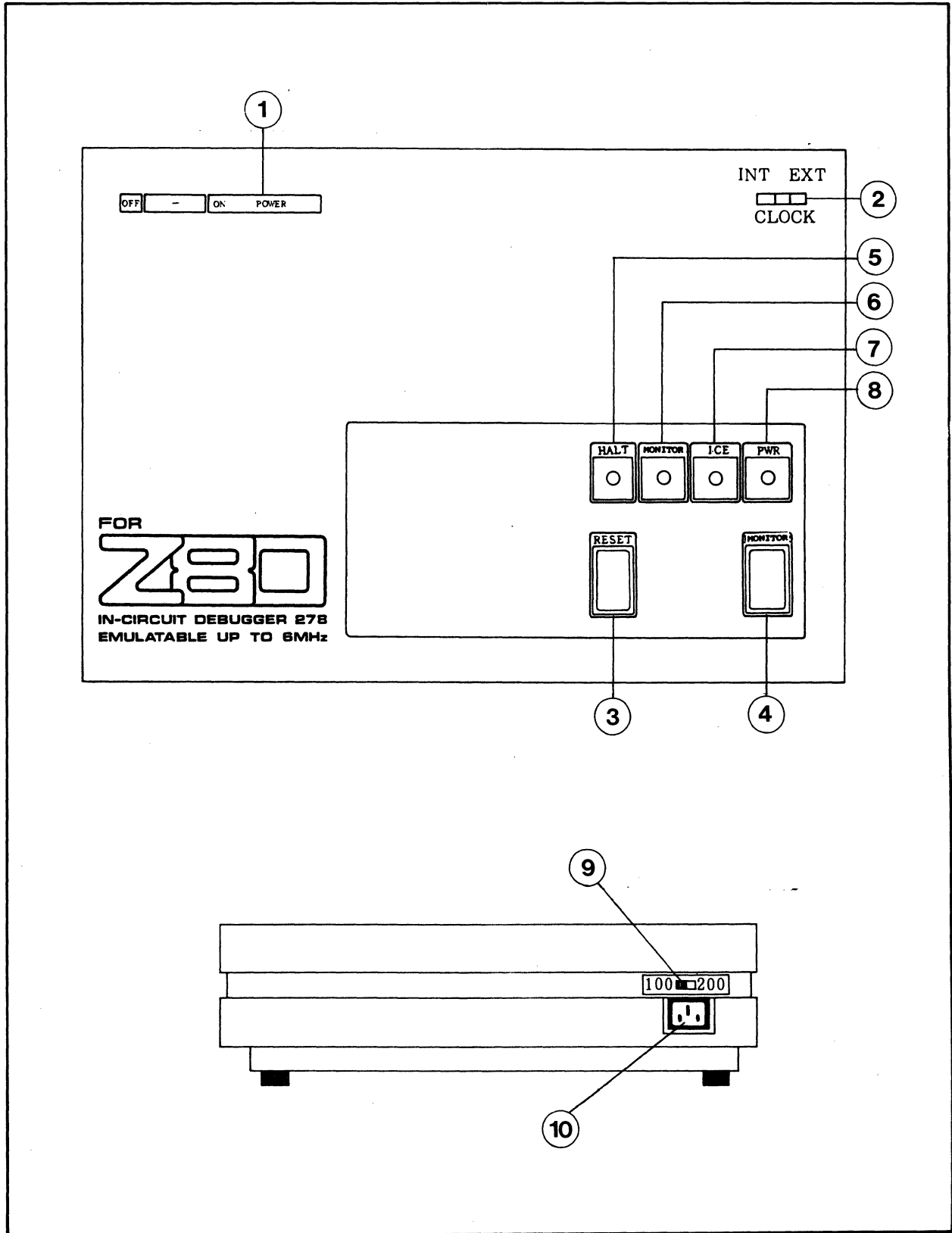


Figure 2-1. ICD Controls and Components

- 10 TERMINAL port connector. This connector attaches a console to the ICD-278. In a standalone configuration (LOCAL), it is used to input or output ICD operator commands. In REMOTE mode it is used for an auxiliary I/O.
- 11 HOST/AUX (Host/Auxiliary) port connector. This connector is used to connect a Host Computer system or auxiliary I/O to the ICD-278. In a standalone configuration (LOCAL), objects, registers or memory dumps interface to a host computer, printer, teletypewriter, etc., thru this port. In REMOTE mode it can be used to input or output ICD operator commands under control of the host computer.
- 12 LOCAL/REM (Local/Remote) select switch. This switch is used to select the ports through which the ICD operator commands are entered.
- 13 DCE/DTE select switch. This switch is used to set the HOST/AUX port to RS232C data terminal equipment(DTE) or data circuit-terminating equipment(DCE). Factory setting is DTE.

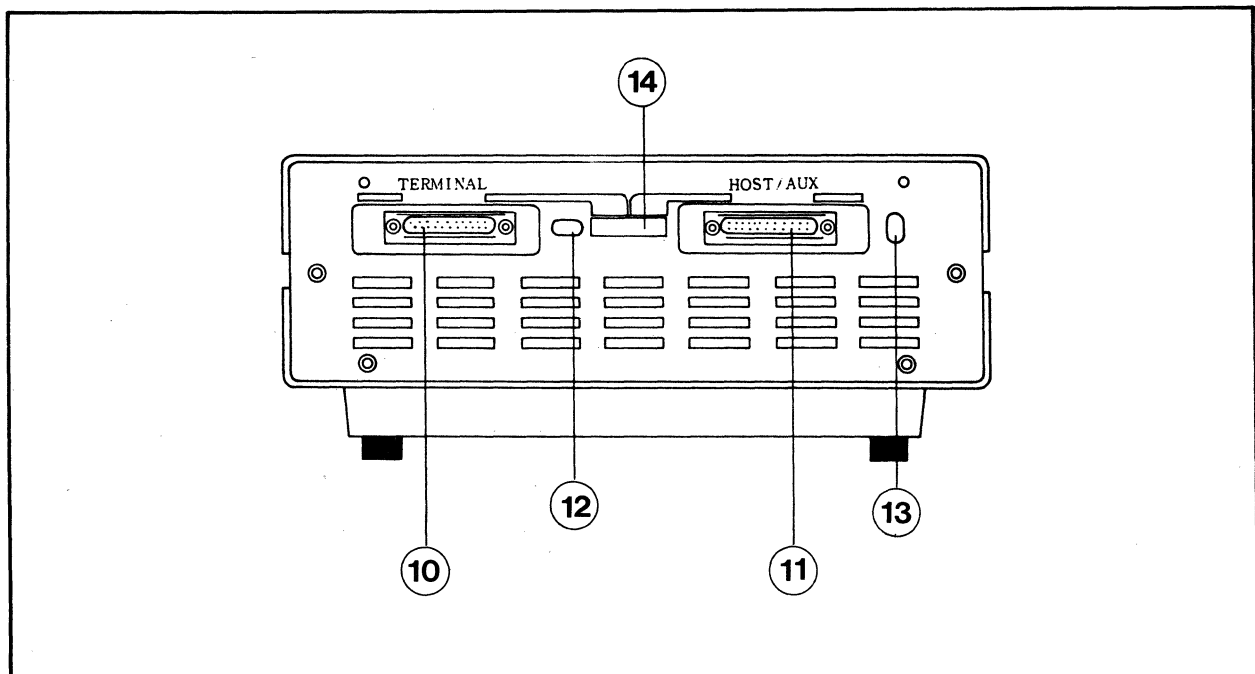


Figure 2-2. ICD Controls and Components

- 14 BAUDRATE set switch. This switch is used to set the baud rates for the TERMINAL and the HOST/AUX port (Table 2-1). To set the baud rate, turn dial using flat-head screwdriver or equivalent.

NOTE: There are 14 usable baud rates available. Baud rate switch numbers E and F are not recommended.

Factory setting = 1(9600bps) for TERMINAL and HOST/AUX ports.

BAUD RATE SET SWITCH NO.	BAUD RATE(BPS)
0	19,200
1	9,600
2	4,800
3	2,400
4	1,200
5	600
6	300
7	150
8	75
9	110
A	134.5
B	200
C	1,800
D	2,000
E	—
F	—

Table 2-1. ICD Baud Rates

15 CPA In-circuit probe plug connector.

16 CPB In-circuit probe plug connector.

NOTE: It is not necessary to connect the in-circuit when running the ICD in the IO mode (in-circuit mode 0) only.

CAUTION: DO NOT REVERSE PLUG PROBE CONNECTIONS. CPA/CPB MISMATCH WILL CAUSE DAMAGE TO THE ICD-278.

17 DB. EMUL (Data Bus Emulator) connector. This connects directly to the data bus of a target system (in-circuit) when decoding the 'RET I' instruction (See Technical References).

18 EXT. BRK. (External Break) connector. This is used to connect an external break cable to the ICD (Table 2-2).

19 EVENT TRG. (Event Trigger) connector. This is used to connect an event trigger cable to the ICD (Table 2-3).

20 E.M.SEL (Emulation Select) switch. This switch is used to set the machine cycle operation of the target system (See Technical References).

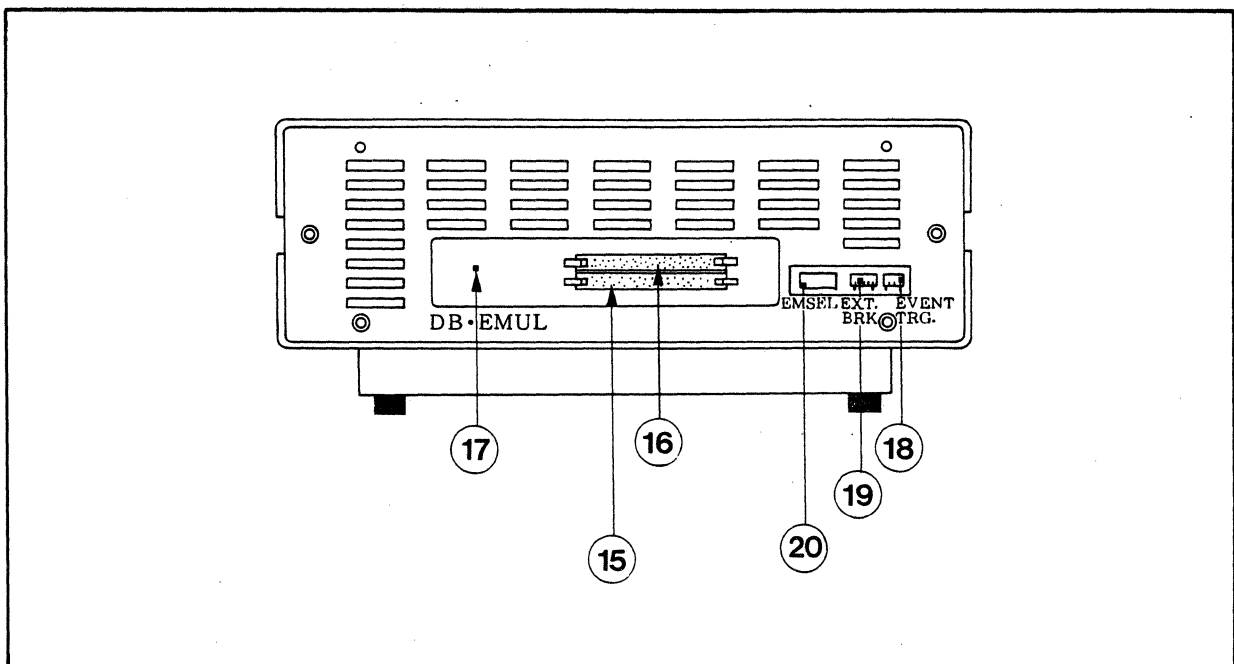


Figure 2-3. ICD Controls and Components

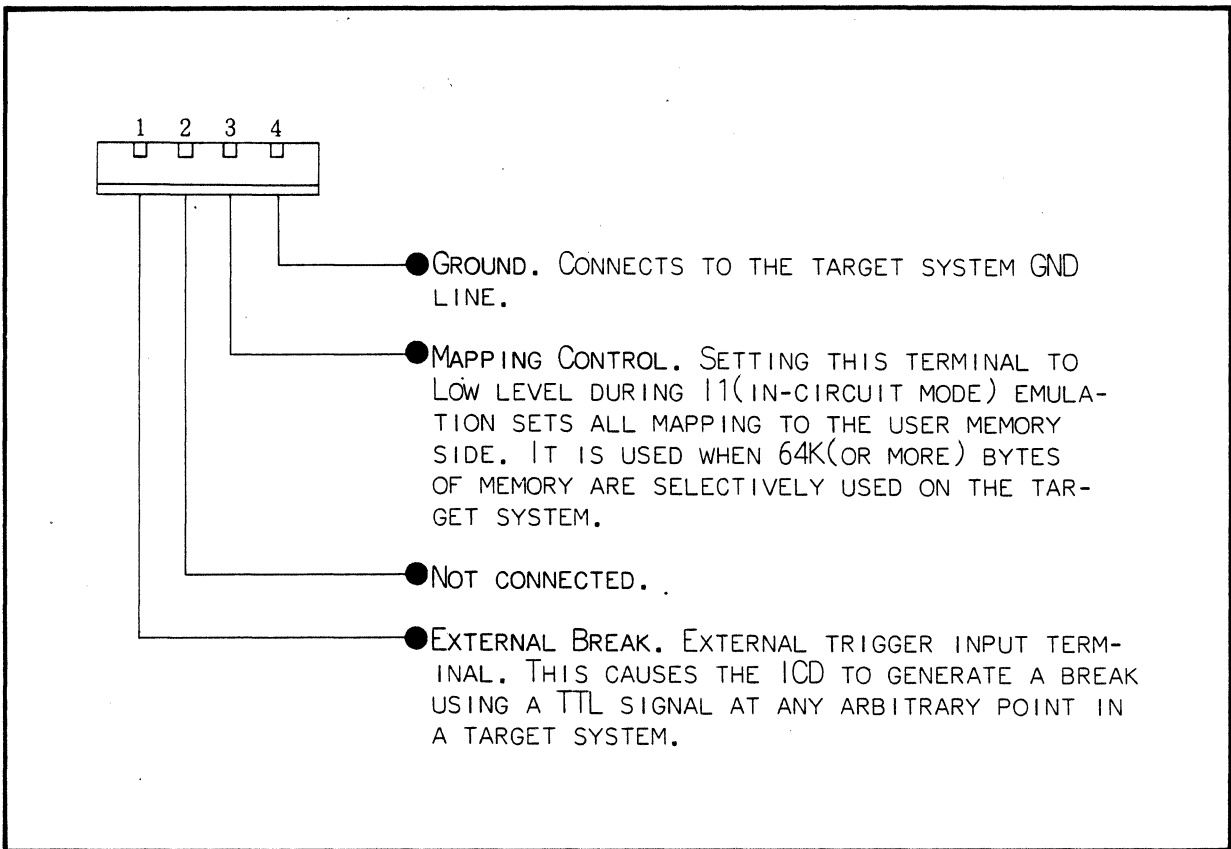


Table 2-2. External Break Connector Diagram

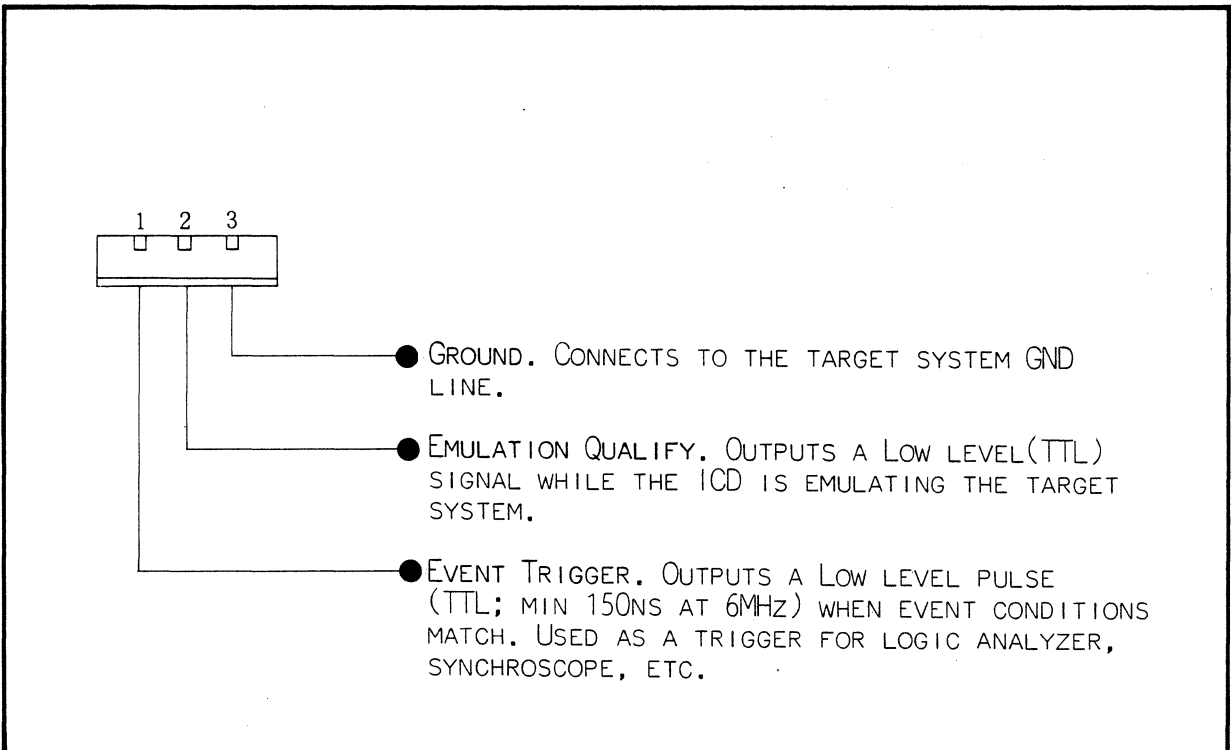


Table 2-3. Event Trigger Connector Diagram

CSA/CSB In-circuit Probe Socket Connector (Figure 2-4)
Connects CPA to CSA and CPB to CSB.

- 21 CSA In-circuit Socket A.
- 22 CSB In-circuit Socket B.
- 23 Pin #1 Plugs into CPU socket. Note proper polarity.
- 24 Polarity mark.
- 25 Pin #1.

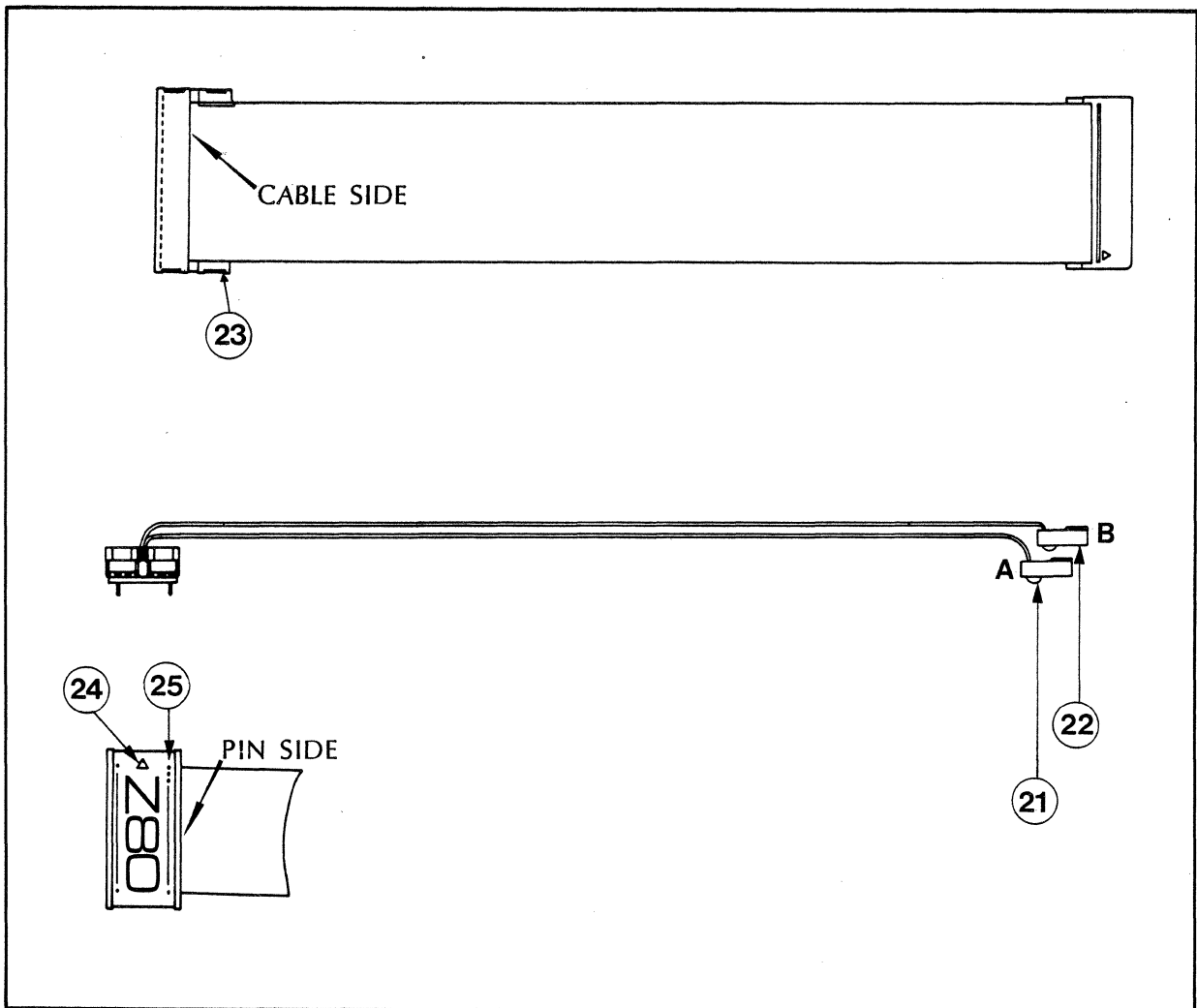


Figure 2-4. CPU Plug Probe

External Break Cable (Figure 2-5). Connects to the EXT. BRK. socket of the unit.

- 26 External Break Probe (red).
- 27 Map Control Probe (yellow).
- 28 Ground clip (black).

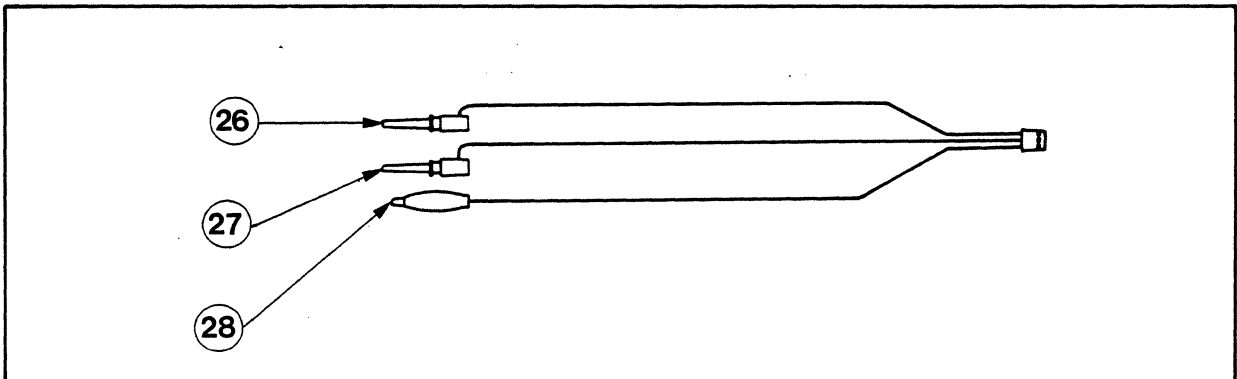


Figure 2-5. External Break Cable

Event Trigger Cable (Figure 2-6). Connects to the Event TRG. socket of the unit.

- 29 Event Trigger Probe (green).
- 30 Emulation Qualifier Probe (white).
- 31 Ground clip (black).

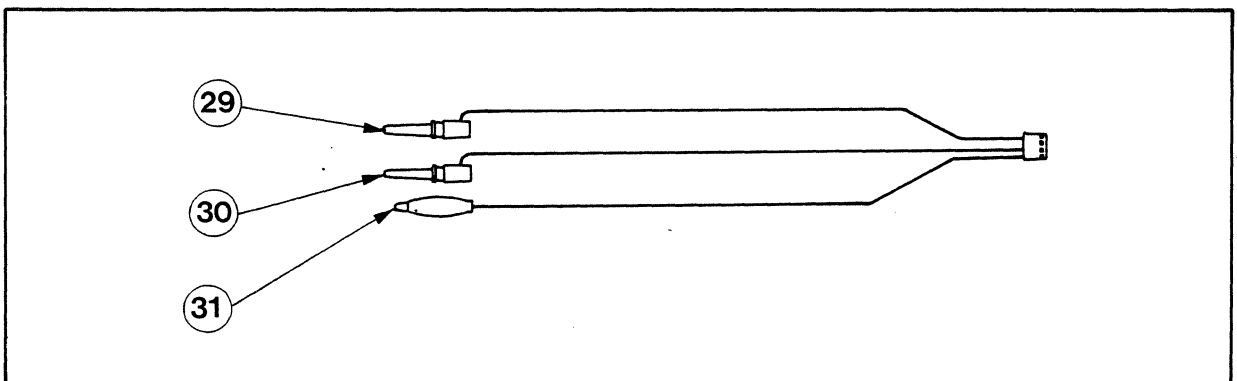
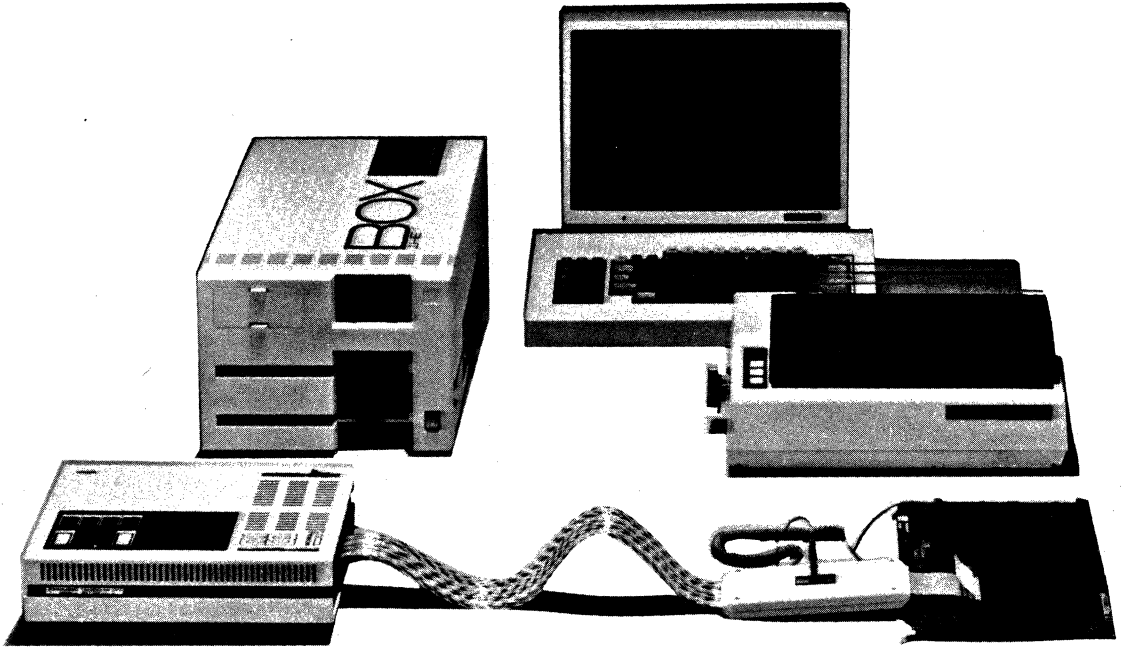


Figure 2-6. Event Trigger Cable



SECTION III

SYSTEM CONFIGURATION / START-UP / COMMAND GUIDE

This section introduces the user to the various ICD system configurations, including communication with host computer systems. Refer to this section when interfacing the ICD-278 to other devices. This sections also shows the correct start-up procedure for the ICD-278.

Additionally, a complete listing of emulator commands and examples are included in this section. Before using the emulator, first construct the the proper system configuration for the ICD-278 and then execute the start-up sequence.

The ICD-278 may be set up to run in one of three different system configurations:

1) LOCAL Basis (standalone configuration)

In this mode the ICD-278 is connected to a console terminal thru which commands may be entered. The Host/Aux port may be used to connect a printer, teletypewriter. Etc. Hardcopy operations may then be produced by issuing a "Print" command.

2) LOCAL Basis (standalone with host computer)

In this mode the ICD-278 performs object file input/output with a host computer. ICD operator commands may be entered from the console terminal. The "User" command enables direct communication between the ICD console terminal and the host computer.

3) REMOTE Basis

In this mode input/output from the host computer is performed by the utility software program; ZICE. The ICD-278 may be furnished with the ZICE command for batch processing and other applications (see Part II, Software Specification Guide).

The three system configurations are illustrated by symbolic notation (Table 3-1). The start-up sequence is then performed (Table 3-2) after constructing the proper system configuration. The ICD-278 is initialized when its mainframe is first powered up or when the RESET switch is pressed after powering up.



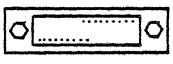
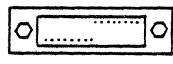
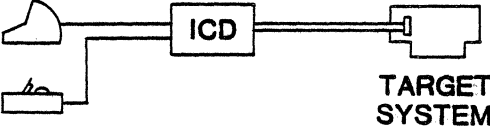


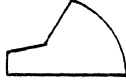
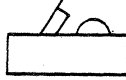
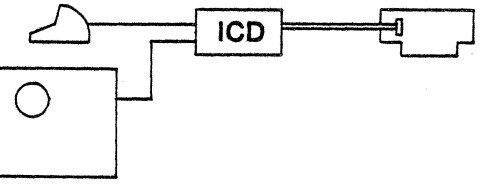




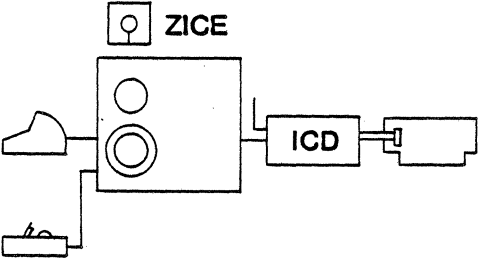


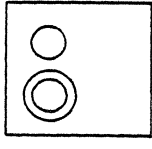
SYSTEM CONFIGURATION ICD	LOCAL  REM	DCE  DTE	TERMINAL 	HOST/AUX 
LOCAL (standalone) 	LOCAL  REM	DCE  DTE	 CONSOLE	 PRINTER
LOCAL (standalone with Host Computer) 	LOCAL  REM	DCE  DTE	 CONSOLE	 HOST SYSTEM
REMOTE 	LOCAL  REM	DCE  DTE		 HOST SYSTEM

Table 3-1. System Configuration

System Configuration Description

Use Table 3-1 to establish the correct system configuration for the LOCAL or REMOTE modes. Before constructing the system make sure that the power to each component is OFF.

First set the LOCAL/REM and DCE/DTE switches to the positions specified in the configuration diagram.

The ICD-278 uses three cables to connect it to various communication devices and the target system. The two blue-colored ribbon cables are used to connect the ICD to; 1) a console terminal through the TERMINAL port connector, and 2) an auxiliary printer or a host computer (Figure 2-2). Connect these cables to the device selected and then to the ICD HOST/AUX or TERMINAL port male sockets.

The CPU Plug Probe cable (Figure 2-4) connects the ICD to the target processor socket. Remove the existing CPU (Z80) from the target system and insert the plug probe (Z80 40-pin end) into the adaptor socket and then carefully insert the assembly into the target system CPU socket. Do not bend the pins.

NOTE: Do not place the cables close to the power supply or interference will result.

Next connect the CSA/CSB Incircuit sockets to the ICD CPA/CPB connectors (Figure 2-3 and 2-4). The cable with the longest length (CSB socket B) MUST be connected to the UPPER (CPB) Incircuit plug probe connector on the ICD.

NOTE: THE CABLE WITH THE LONGEST LENGTH (CSB) MUST BE CONNECTED TO THE UPPER INCIRCUIT PLUG PROBE CONNECTOR (CPB).

CAUTION: DO NOT REVERSE PLUG PROBE CONNECTIONS. MISMATCH OF CSA/CSB AND CPA/CPB WILL CAUSE SEVERE DAMAGE TO THE ICD-278.

Once the system is constructed, refer to the ICD start-up sequence illustrated in Table 3-2.

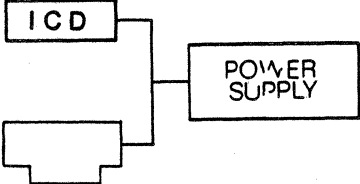




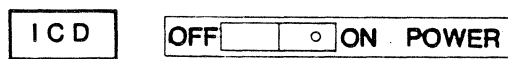
<p>1</p> 	<p>Connect the ICD and target system to the same primary power supply. Be sure all power is OFF before connecting components.</p>
<p>2</p> 	<p>Select the proper voltage requirement by setting the POWER select switch located on the ICD mainframe.</p>
<p>3</p> 	<p>Set the CLOCK switch located on the ICD Operator Panel to either EXT (External/Target clock) or INT (Internal/ICD clock).</p>
<p>4</p> 	<p>Turn the target system power ON.</p>
<p>5</p> 	<p>Turn the console terminal power ON.</p>
<p>6</p> 	<p>Turn the ICD power ON. A response message will then be displayed on the console terminal, after which the ICD will wait for a command entry.</p>

Table 3-2. ICD Start-up Sequence

COMMAND GUIDE

The information contained in this COMMAND GUIDE relates to the ZAX ICD-278 for Z80 series emulators.

This Command Guide contains two parts. Part one acts as a reference guide and details the format and parameters of each command. Part two is a quick-reference guide that omits the parameter explanations for the experienced operator.

The ICD-278 for Z80 is capable of executing 29 major software commands. A portion of these commands are divided into various sub-groups for a total of 61 different command formats.

*Example of command format structure:

```
1 History format display - 2 displays in disassembled
  or machine cycle format;

3
● >H M|D , int.point , term.point (cr)

M 4           :display in machine cycle format.
D 7           :display with assembled codes.

int.point    5           :display or search initiation storage
                    pointer [decimal 1 to 2047] 6
term.point   :display or search termination storage
                    pointer [decimal 1 (default to 2047)] 6
```

Where -

- 1 - Command name.
- 2 - Command description.
- 3 - Command format including available parameters.
- 4 - User supplied keywords, showing input choices.
- 5 - User supplied address/data keywords.
- 6 - Address/data parameters.
- 7 - Keyword/address/data explanations.

Z80 Command Reference Guide -

Commands: 61

Assemble

Break status

Hardware Break specification

Hardware Break designation

Software Break specification

Software Break designation

Software Break op code specification

External Break hi/lo designation

External Break on/off designation

Event Break designation

Timeout Break designation

Write protect Break designation

Hardware arm/individual designation

Break initialize

Break event with passcount

Calculation

Compare

Disassemble

Dump

Event status

Event specification

Event designation

Examine only

Examine and change

Fill

Go

History - realtime trace status

History - realtime trace counter reset

History - monitor trigger storage

History - event trigger

History - format display

History - search

Identification

Incircuit status

Incircuit specification

Load

Move

Mapping status

Mapping specification

Next

Offset status

Offset specification

Pin status

Pin specification

Port examination

Port examination and change

Print

Quit

Register status

Register reset

Register change

Save

Search

Supervisor status

Supervisor specification

Trace status

Trace designation

Trace specification

User

Verify

Zice

Command Notes:

*Keywords/key characters are displayed as; M / , etc, or as BOLD characters. They represent words, letters, and characters which must be entered. Any combination of upper/lower case letters may be used.

*Lowercase letters show items which the user must supply, such as "filename" in which the user would enter the name of a selected file.

*A vertical line "|" is shown to indicate a choice between separated information which must be selected, such as;

ON|OFF

ON or OFF may be entered, but not both.

*Include all punctuation such as commas, equal signs, colons, slashes, feature keys. Etc.

*Available address/data parameters are given at the end of the explanations and grouped as "[hex16/hexdata/L byte/ASCII 32, etc]."

These address/data parameters include;

[hex16]	:hexadecimal 16-bit memory address.
[hexdata]	:word or byte data (8/16 bit). Valid range is 0-F and X.
[L byte]	:number of bytes. Hexadecimal 16-bit, 0-F.
[ASCII 32]	:ASCII codes. All ASCII A thru Z characters(32) are available.
decimal	:decimal values are specified within the brackets.

*Commands are divided into different sub-groups based on their operation. This is shown whenever one command can perform several different functions, such as the Event Command. In this case such sub-groups as Status, Designation and Specification are used.

Status means; COMMAND name (cr).
Designation means; COMMAND name ON/OFF or ON/OFF/CLR.
Specification means; COMMAND name address, data, etc.

Other sub-groups are individually identified by selective titles (i.e. COMMAND: change, reset, display, etc).

INTERRUPTION Process

Emulation stop
with GO command :press MONITOR swith on Operator Panel.

Stop :pressing <ESC> key on console keyboard
stops the corresponding command.

Interrupt <SP> :pressing the (space) bar on the console
keyboard interrupts sequence after one
line is displayed on screen. To restart
display, press (space) bar again.

<DC3> :pressing the (CTR-S) key on the console
keyboard immediatly interrupts display.

<DC1> :pressing the (CTL-Q) key on the console
keyboard restarts the interruption.

INPUT STATEMENT CORRECTION

Correction of
one character :(delete) Data is cancelled by the
number of Rub out specifications and
the corresponding characters are dis-
played.

Backspace :<BS>(backspace) Backspace operation is
performed by the specified number and
corresponding data cancelled.

Cancellation of
all data entry :<NAK> Pressing (CTL-U) cancels all
entry data.

Error Messages

ERROR MESSAGE	COMMAND OCCURRENCE	DISPLAYED WHEN;
UNABLE BREAK ADDRESS	Break Go	a software break is specified in the non-RAM area
MULTI BREAK ADDRESS	Break Go	a software break is duplicated at the same address
WARNING UNABLE SOFT BREAK	Break	a software break is set at the address presently not mapped in RAM
*** FILE NOT FOUND	All	no file exists
*** DISK READ ERROR	All	a disk read error occurs
*** DISK WRITE ERROR	All	a sum check error occurs
*** NO DIRECTORY SPACE	All	no blank area available
C?>	All	a command code error occurs
P?>	All	a parameter code error occurs
/?>	All	a modifier code error occurs
MEMORY WRITE ERROR AT XXX	Assemble, Examine Load, Move, Fill	there is a memory modification error.
MEMORY TIMEOUT ERROR AT XXXX	Assemble, Dump, Fill Examine, Load, Move	memory I/O in the target system does not respond to ICD access.
I/O TIMEOUT ERROR AT XX	Port, Verify, Save Compare, Search	timeouts a waitstate
XXXX INPUT ERROR	Examine, Assemble, Port	as an input error
BREAK BUSY	Break, Go	the break specification exceeds the limit

THIS PAGE INTENTIONALLY BLANK

Commands

ASSEMBLE specification - changes Z80 assemble codes to Z80 machine codes and modifies the storage contents;

●>A mem.addr (cr)

xxxx (Z80 assemble code) (cr)
xxxx (cr)

mem.addr :memory address [hex16].
xxxx :ICD displays address then input Z80
assemble code.

EXAMPLE:

MOVE THE DATA THAT RESIDES IN
THE TARGET SYSTEM MEMORY
0 - 0FFF, TO THE EMULATOR
MEMORY. START AT THE ICD
ADDRESS, 0;

>M 0, FFF,0,UP

BREAK status - displays the existing break status;

●>B (cr)

Hardware BREAK specification - sets a hardware break;

●>B/ A|B|C stat , addr , passcount (cr)

A|B|C :hardware break name.

stat >>> break status;

M :memory access
P :port access
MR :memory read
MW :memory write
PW :port write
PR :port read
OF :operation code fetch
IA :interrupt acknowledge

addr :break address [hex16].

passcount :break passing count [decimal 1 (default)
to 65535].

Hardware BREAK designation - turns on, off or clears a specified hardware breakpoint;

●>B/ A|B|C ON|OFF|CLR (cr)

A|B|C :hardware breakpoint name.

Software BREAK specification - sets a software break;

●>B/ 0-7 addr , passcount (cr)

0-7 :software break name

addr :break address [hex16].

passcount :specifies break passing count (loop counter) [decimal 1 (default) to 65535].

Software BREAK designation - turns on, off or clears a specified software break;

●>B/ 0-7 ON|OFF|CLR (cr)

0-7 :software break name

Software BREAK op code specification - specifies to place a software break in the enable (EN) state or the disabled (DI) state;

●>B S= EN|DI (cr)

External BREAK hi/low designation - specifies an external break to occur at the HI or LOw edge;

●>B/X HI|LO (cr)

External BREAK designation - turns on, off or clears an external break;

●>B/X ON|OFF|CLR (cr)

Event BREAK designation - turns an event break on or off;

●>B/E ON|OFF (cr)

Timeout BREAK designation - turns a timeout break on or off;

●>B/T ON|OFF (cr)

Write protect BREAK designation - turns a write protect break on or off;

●>B/W ON|OFF (cr)

Hardware arm/individual BREAK specification - used with a specified hardware break name (A,B or C). Arm enables a break to occur after an event trigger takes place. INDividual allows a break to occur without regard to an event trigger;

●>B/ A|B|C ARM|IND (cr)

A|B|C :hardware break name

BREAK initialize - clears the event passing condition and resets the ARM specification;

●>B INI (cr)

BREAK event with passcount - specifies the number of software pass counts;

●>B/E passcount (cr)

passcount :specifies break passing count [decimal 1 (default) to 65535].

EXAMPLES (BREAK command):

DISPLAY BREAK STATUS WITHOUT HARDWARE BREAK ON;

```
>B
T (ON)
S (DI)
W (OFF)
```

DISPLAY A BREAK STATUS WITH HARDWARE BREAK ON;

```
>B OF,100
>
>B
A (ON) OF 0100 1 0 IND (0000_0001_0000_0000)
T (ON)
S (DI)
W (OFF)
```

SET BREAK STATUS WITH A 16-BIT PHYSICAL ADDRESS AND ISSUE A BREAK STATUS COMMAND;

```
>B/A M,000X_111X_XXX_0000
>B
A (ON) M XXX0 1 0 IND (000X_111X_XXX_0000)
T (ON)
S (DI)
W (OFF)
```

SET A SOFTWARE BREAK USING A HEXADECIMAL ADDRESS;

```
>B/7 1234
>B
A (ON) M XXX0 1 0 IND (000X_111X_XXX_0000)
7 (ON) 1234 1 0
T (ON)
S (DI)
W (OFF)
```

SET ARM BREAK FOR HARDWARE BREAK A;

```
>B/A ARM
>B
A (ON) M XXX0 1 0 ARM (000X_111X_XXX_0000)
7 (ON) 1234
T (ON)
S (DI)
W (OFF)
```

ENABLE A SOFTWARE BREAK;

```
>B S=EN
>B
A (ON) M XXX0 1 0 ARM (000X_111X_XXX_0000)
7 (ON) 1234
T (ON)
S (EN)
W (OFF)
```

CALCULATION - allows subtraction and addition of hex and decimal numbers. Both subtraction and addition operations can be performed on the same line.

●>C data ± data ± data ... (cr)

data :operation data. Decimal valid for -8388608 to 8388607, hex valid for 0 to FFFFFFF(H).

EXAMPLE:

EXECUTE MULTIPLE HANDLING OF
SUBTRACTION AND ADDITION USING
HEX AND DECIMAL DATA
INTERCHANGABLY;

>C 1234+1234
H: 0009A4
D: 2468

>C 1000H-FFFH
H: 000001
D: 1

COMPARE - compares the contents of a specified memory and displays the unmatched data;

●>CO beg.addr , end addr , comp.addr , UP|PU (cr)

beg.addr :comparison beginning address [hex16].
end addr :ending comparison address [hex16/L byte].
comp.addr :memory address to be compared [hex16].

UP :use beg.addr for the user memory and comp.addr for the ICD program memory.
PU :use beg.addr for the ICD program memory and comp.addr for the user memory.

EXAMPLE:

MEMORY COMPARISON OF THE RANGE; 0 TO 0001

>CO 0,FFF,0001,UP <CR>

1	2	3	4
ADDRESS	M	ADDRESS	M
0000	C3	0000	FF
0001	06	0001	8F

●NOTE:

- 1 - TARGET SYS MEMORY ADDRESS
- 2 - TARGET SYS MEMORY CONTENTS
- 3 - EMULATOR MEMORY ADDRESS
- 4 - EMULATOR MEMORY CONTENTS

DISASSEMBLE - disassembles the memory contents and displays it;

●>DI beg.addr , end addr (cr)

beg.addr :beginning memory address [hex16].
end addr :ending memory address [hex16/L byte].

EXAMPLE:

```
>DI 100
0100 2100B0 LD HL, B000H
0103 1100A0 LD DE, AAAAH
0106 7E LD A,(HL)
0107 B7 OR A
0108 C20003 JP NZ,0300H
010B 12 LD (DE),A
010C 23 INC HL
010D 13 INC DE
010E C30601 JP 0106H

>DI 100,114
0100 2100B0 LD HL,0B000H
0103 1100A0 LD DE,0A000H
0106 7E LD A,(HL)
0107 B7 OR A
0108 C20003 JP NZ,0300H
010B 12 LD (DE),A
010C 23 INC HL
010D 13 INC DE
010E C30601 JP 0106H
0111 00 NOP
0112 3A003A LD A,(3A00H)
```

DUMP - displays the memory contents in either byte or word units. The contents is represented by a hex number or ASCII code;

●>D/ W beg.addr , end addr (cr)

W :dump memory contents on a word basis (default is a byte basis).

beg.addr :beginning memory address [hex16].
end.addr :ending memory address [hex16/L byte].

EXAMPLE:

DUMP MEMORY CONTENTS FROM 0 TO 30;

```
>D 0,30
0 1 2 3 4 5 6 7 8 9 A B C D E F ASCII CODE
0000 21 FF 00 11 34 12 77 12 13 23 B7 C2 06 00 C3 00 !...4.w..#7B..C.
0010 01 00 00 00 .....
0020 00 00 0.....
0030 00
```

DUMP MEMORY CONTENTS INTO WORD;

```
>D/W A000,L40
0 2 4 6 8 A C E ASCII CODE
A000 F07F A18F F805 900F F01D AFAF F00D 001C .P.!X...P//.P..
A010 F017 E5AF F005 240F F005 BDAF F0ED 180F .P/E.P.$P/=MP..
A020 D00 84AF F009 010F D21F A74F F025 021F .P/..P...RO'%P..
A030 F00D A125 F005 000F F30F A7AF F045 520F .P%!..P...S/'EP.R
```

EVENT status - displays the current event status;

●>EV (cr)

EVENT specification - sets or releases an event trigger;

●>EVST=stat A=addr D=data (cr)

stat >>> event trigger status;

M :memory access
MR :memory read
MW :memory write
P :port access
PR :port read
PW :port write
OP :operation code fetch
IA :interrupt acknowledge

addr :directs event trigger at a specified break
address [hex16].

data :specifies event data [hexdata].

EVENT designation - turns on, off or clears an event
trigger;

●>EV ON|OFF|CLR (cr)

EXAMPLES (EVENT command):

SPECIFY AN EVENT HEX ADDRESS USING 'DON'T CARE';

```
>EV A=AXX0  
>EV  
(ON)  
STATUS = ANY  
ADDRESS = AXX0 (1010_XXXX_XXXX_0000)  
DATA = XX (XXXX_XXXX)
```

SPECIFY EVENT DATA;

```
>EV D=0010_11XX  
>EV  
(ON)  
STATUS = OF  
ADDRESS = AXX0 (1010_XXXX_XXXX_0000)  
DATA = 2X (0010_11XX)
```

SET EVENT COMMAND USING ADDRESS, DATA AND STATUS;

```
>EV A=00FF D=12 ST=MR  
>EV  
(ON)  
STATUS = MR  
ADDRESS = 00FF (000_0000_1111_1111)  
DATA = 12 (0001_001)
```

EXAMINE only - examines the memory contents and displays it in either ASCII or hex data;

●>E/ W addr (cr)

W :examine memory contents on a word basis (the default is a byte basis).

addr :beginning address of examined memory [hex16].

EXAMINE and change - examines and changes the memory contents to either ASCII or hex data;

●>E/ W beg.addr = mod.data (cr)

W :change memory contents on a word basis (the default is a byte basis).

beg.addr :beginning memory address to be changed [hex16].

mod.data :memory modification data [hexdata/ASCII32].

note: multi-display of data;

xxxx xx=data (cr) :displays data at next address after change.
,(cr) :displays data at same address after change.
^(cr) :displays data at address decremented by 1 after change.
/(cr) :terminates Examine command after change.

When data is omitted, memory contents remain unchanged.

EXAMPLE:

EXAMINE AND CHANGE MEMORY IN
BYTE DATA;

>E 0100 <CR>
0100 FA = CHANGE DATA <CR>
0101 03 =
0102 21 =
0103 1F =
0104 FD =
0105 11 =
0106 AB =
0107 FD =

EXAMINE AND CHANGE MEMORY IN
WORD DATA;

>E/W 100 <CR>
0100 03FA = CHANGE DATA <CR>
0102 1F21 =
0104 11FD =
0106 FDAB =
0108 200E =
010A FECD =

FILL - used to fill memory with either hex or ASCII codes;

●>F W beg.addr , end addr , data (cr)

W :fill memory contents on a word basis (the default is a byte basis).
beg.addr :beginning address to be filled [hex16].
end addr :end memory address [hex16/L byte].
(default = data to be filled from the beginning address is written).
data :specifies data to be filled [hexdata/ASCII32].

EXAMPLE:

FILL MEMORY WITH "0" FROM
ADDRESS 0000 TO 00FF;
>F 000,00FF,0

FILL MEMORY WITH ASCII
CHARACTER FROM 0 TO FF;
F 000, 0FF, 'ZAX'

GO - executes the user program;

●>G beg.addr , end addr , end addr* (cr)

beg.addr :beginning executable address [hex16].
end addr :ending executable address [hex16].
end addr* :optional second end address [hex16].

EXAMPLE:

EXECUTE A USER PROGRAM STARTING AT 0H;

>G 0,B

PC	MC	OP	SP	AF	BC	DE	HL	IX	IY	I	IF(SP)
000B	1100B0	LD DE,0B0000H	FFFC	0000	0000	B000	A000	0000	0000	00	0 0400
<BREAK HARDWARE A>											

CONTINUE THE EXECUTION AFTER BREAK;

>G

PC	MC	OP	SP	AF	BC	DE	HL	IX	IY	I	IF(SP)
15DC	3A003A	LD A,(3A00H)	FFB6	3A00	1410	B001	9F01	0000	0000	00	0 0300
<BREAK MONITOR>											

HISTORY - implements the trigger modes (6 types) for realtime tracing, and displays or searches for the user program operation to be traced.

Realtime trace status (HISTORY) - displays the current trace status;

●>H (cr)

Realtime trace counter reset (HISTORY) - clears (resets) the realtime trace counter;

●>H CLR (cr)

Monitor trigger storage (HISTORY) - specifies the begin, end, center or multiple monitor trigger;

●>H BM|EM|CE|ME , range (cr)

BM :begin monitor trigger storage. Trace section is initiated by emulation start and terminated at a predetermined breakpoint specified by the range.

EM :end monitor trigger storage. After initiating emulation with the Go or Next command, the monitor will respond with a break when the trace is terminated. Maximum range is 2048.

CE :center event trigger. Trace section is recognized both before and after the event point.

ME :multiple event trigger. Trace is performed each time an event point is passed during loop processing.

range :trace range [decimal 1 to 2047]

End trigger (HISTORY) - sets the end monitor or end event trace specification;

●>H EM|EE (cr)

EM :end monitor trigger. Emulation begins using the GO or NEXT command to start trace and ends when control returns to monitor by passing a breakpoint.
EE :end event. Terminates trace after event point is passed. Trace range is 2047.

HISTORY format display - displays in disassembled or machine cycle format;

●>H M|D , int.point , term.point (cr)

M :display in machine cycle format.
D :display data with assemble codes.
int.point :display or search initiation storage pointer [decimal 1 to 2047].
term.point :display or search termination storage pointer [decimal 1 (default) to 2047].

HISTORY search - searches for the contents of the real time trace buffer;

●>HS,./addr/data/cycle , int.point , term.point (cr)

addr :search address [hex] (word address="addr W")
data :search data [hexdata].

cycle >>> specifies the type of machine cycle;
IA :interrupt acknowledge
MR :memory read
MW :memory write
PR :port read
PW :port write
OF :operation code fetch
HA :halt acknowledge

int.point :search initiation storage pointer [decimal 1 to 2047].

term.point :search termination storage pointer [decimal 1 (default) to 2047].

EXAMPLES (HISTORY command):

DISPLAY THE HISTORY STATUS;

```
>H
CLOCK COUNTER = 000000AA/
STORAGE MODE = EM
STORAGE SIZE = 14/ 14
```

DISPLAY THE REAL TIME TRACE IN MACHINE CYCLE;

```
>HM,20
POINT ADDR DT ST
0020 0003 11 M1
0019 0004 00 MR
0018 0005 04 MR
0017 0006 E5 M1
0016 0005 03 MW
0015 0004 00 MW
0014 0007 D5 M1
0013 0003 04 MW
0012 0002 00 MW
0011 0008 21 M1
0010 0009 00 MR
0009 000A A0 MR
0008 000B 11 M1
0007 000C 00 MR
0006 000D B0 MR
0005 000E 7E M1
0004 A000 3A MR
0003 000F 77 M1
0002 A000 3A MW
```

DISPLAY THE REALTIME TRACE IN MACHINE CYCLE AND DISASSEMBLE CODE;

```
>H D
POINT ADDR DT ST OP
0023 0000 210003 LD HL,0300H
0020 0003 110004 LD DE,0400H
0017 0006 E5 PUSH HL
0016 0005 03 MW
0015 0004 00 MW
0014 0007 D5 LD HL,0A000H
0013 0003 04 MW
0012 0002 00 MW
0011 0008 2100A0 LD HL,0A000H
0008 000B 1100B0 LD DE,0E000H
0005 000E 7E LD A,(HL)
0004 A000 3A MR
0003 000F 77 LD (HL).A
0002 A000 3A MW
```

SEARCH FOR THE CONTENTS OF REAL TIME TRACE FOR A MEMORY READ;

```
>H S,///MR
POINT T ADDR DT ST
0013 0101 00 MR
0012 0102 A0 MR
0010 0104 00 MR
0009 0105 B0 MR
0007 A000 FF MR
```

IDENTIFICATION - displays an ICD device name and a version of the installed firmware;

●>ID (cr)

INCIRCUIT status - displays the current incircuit mode;

●>I (cr)

INCIRCUIT specification - sets the ICD mapping mode;

●>I 0|1|2 (cr)

0 :System mode enables debugging (software only) using the ICD program memory. In this mode the target system I/O and interrupt signals are ignored.

1 :Partial mode. Enables debugging using the ICD program and target system memories. In this mode the specified mapping, I/O and interrupt signals all become valid.

2 :All mode. Enables debugging using only the target system memory. R/W and RO memories are operated as user memory (US).

note: default = current state displayed

LOAD - loads either a hex file on diskette or the object program (Intel format) from the specified port;

●>L/ T|P|A|H filename , bias (cr)

T :terminal port (ignore software handshake)
P :terminal port (perform software handshake)
A :auxiliary port
H :host port

filename :(available with ZICE software only).

bias :if omitted, starting address is specified [L byte].

EXAMPLE:

LOAD THE TEST HEX FILE ON
DISKETTE;

> L TEST
100

>L/T ,100
200

MOVE - moves memory between the ICD and target system;

●>M beg.addr , end addr , mov.addr , UP|PU (cr)

beg.addr :beginning memory address [hex16].
end addr :ending memory address [hex16/L byte].
mov.addr :beginning address where data is transferred to [hex16].

UP|PU >>> optional parameter;

UP :data is moved from the target memory to the ICD program memory.
PU :data is moved from the ICD program memory to the target memory.

EXAMPLE:

MOVE MEMORY OF 0H THROUGH
FFFH;

>M 0,FFF,0,UP

MAPPING status - displays the current map status;

●>MA (cr)

MAPPING specification - sets the ICD memory map. The target system or ICD program memory is specified on a 1K byte basis;

●>MA beg.addr , end addr = RO|RW|NO|US (cr)

beg.addr :beginning address of mapping [hex16].
end addr :ending address of mapping [hex16/L byte]
 note: default (beg.addr/end addr)=1K byte.

RO :read only memory
RW :read/write memory
NO :non memory area (a break occurs in this
 area if accessed by a program)
US :user (target system) memory

EXAMPLE:

SPECIFY 0 TO FFF AS THE USER
MEMORY;

>MA 0,FFF = US

DISPLAY THE STATUS OF THE
MEMORY MAP;

>MA
IN-CIRCUIT MODE 1 (US=RW)
~~0000~~-0FFF = US
~~1000~~-FFFF = RW

DISPLAY THE STATUS OF THE
MEMORY MAP WHEN THE INCIRCUIT
MODE IS NOT 1;

>MA
IN-CIRCUIT MODE 0 (US=RW)
~~0000~~-0FFF = US
~~1000~~-FFFF = RW

NEXT - executes n-step All trace display from the current pointer;

●>N steps (cr)

steps :specifies number of single steps [decimal 1 (default=1) to 15535].

EXAMPLE:

PERFORM A SINGLE STEP TRACE, FIVE STEPS FROM THE CURRENT PC;

>N 5

PC	MC	OP	SP	AF	BC	DE	HL	IX	IY	I	IF(SP)
010	3A00A0	LD A,(0A000H)	002E	050E	A50E	0301	4312	0000	0000	00	0 001A
0103	77	LD (HL),A
0104	FE05	CP 5	0542
0106	C20001	JP NZ,0100H
0109	2100A0	LD HL,0A000H	A000

PERFORM A SINGLE STEP TRACE FOR FIVE STEPS CHANGING PC;

>R PC,200

>N 5

PC	MC	OP	SP	AF	BC	DE	HL	IX	IY	I	IF(SP)
0200	3A003A	LD A,(3A00H)	002E	3A42	A50E	0301	A000	0000	0000	00	0 0000
0203	00	NOP
0204	3A003A	LD A,(3A00H)
0207	00	NOP
0208	3A003A	LD A,(3A00H)

 OFFSET status - displays the current offset set value;

●>O (cr)

 OFFSET specification - sets a value (1 of 4) in an offset register for relative addressing. All offset registers can be used for ICD memory addressing parameters;

●>O &1|&2|&3|&4 = addr (cr)

addr :offset register set value [hex16]

PIN status - displays the current pin status;

●>PI (cr)

PIN specification - masks or unmasks the target system
interruption signal when the incircuit mode is I1.
Interruptions are ignored in the I1 mode and accepted
in the I2 mode.

●>PI level=EN|DI (cr)

level >>> specifies bus request or interrupt level;

BUSRQ :bus request
NMI :non-maskable
INTR :interrupt

EN :interruption or hold signal enable.
DI :interruption or hold signal disable.

note: default = current specification and mask states
are displayed.

EXAMPLE:

DISPLAY PIN STATUS;

```
>PI
IN-CIRCUIT MODE 1
NMI (EN) = 0
BUSRQ (EN)
INTR (EN) = 0
```

DISABLE BUSREQUEST AND DISPLAY
THE STATUS;

```
>PI BUSRQ = DI
>PI
IN-CIRCUIT MODE 1
NMI (EN) = 0
BUSRQ (DI)
INTR (EN) = 0
```


PORT examination - examines the current port contents;

●>P port (cr)

port :conversion start port address [hex16/Lbyte].

PORT examination and change;

●>P port = data (cr)

port :conversion start port address [hex16/Lbyte].

data :port conversion data [hex16/ASCII 32].

note: multi-display of data is possible using the formats below;

xxxx xx : data (cr) - display data at next address
after change.
,(cr) - display data at same address
after change.
(cr) - display data at address that
is decremented by 1 after the
change.
/(cr) - terminates Examine command
after change.

EXAMPLE:

EXAMINE AND WRITE OUT A PORT;

>P FF=12 12 34 56 78

READ AND CHANGE PORT IN BYTES
USING MULTI-DISPLAY FORMAT;

>P 55
55 55:12
56 56:23
57 57:'B'
58 58:'D',
58 58:00^
59 59:12
58 58:/

READ AND CHANGE PORT IN WORDS
USING MULTI-DISPLAY FORMAT;

>P 23
23 23:00
24 24:12
25 25:33
26 26:/

PRINT - outputs data sent from the ICD to the Terminal or Host/Aux ports simultaneously.

●>PR ON|OFF (cr)

ON :output data to both Terminal and Host/Aux ports.

OFF :output data to the console only.

QUIT - returns control to the host system (valid when used with ZICE software only);

●>Q (cr)

REGISTER status - displays the current register status;

●>R (cr)

EXAMPLE:

DISPLAY REGISTER WITH TITLES;

>R

PC	SP	SZHPNC	A	BC	DE	I	IF	(SP)(HL)
0000	0000	000000	00	0000	0000		00	0	0021 21

REGISTER RESET - initializes all registers to 0 by pushing the RESET switch on the Operator Panel;

●>R (push)RESET (cr)

REGISTER change - changes the Z80 CPU register and program counter;

●>R reg = data (cr)

reg >>> register name;

A F B C D E H L I
A' F' B' C' D' E' H' L' I'
BC DH HL IX IY SP
BC' DH' HL'
CY S Z HC P N IF(IFFI)

data :register change data [hexdata].

EXAMPLE:

CHANGE HL TO A VALUE OF A000;

>R HL=A000
>R

PC	SP	SZHPNC	A	BC	DE	I	IF	(SP)(HL)
0000	0000	000000	00	0000	0000		00	0	0021 3A

SAVE - saves a user program on diskette (HEX file) or dumps the user program to a specified port;

●>SA/ T|P|A|H filename,beg.addr,end addr,user addr, (cr)

T :terminal port (ignore software handshake)
P :terminal port (perform software handshake)
A :auxiliary port
H :host port

filename :(available with ZICE software only).

beg.addr :beginning memory address [hex16].
end addr :ending memory address [hex16/L byte].
user addr :beginning user program address [hex16].

EXAMPLE:

GENERATE THE TEST.HEX FILE ON DISKETTE;

>SA TEST,0,37FF,0

SEARCH - searches for the memory contents and displays the matched or unmatched address;

●>S/W /D beg.addr , end addr , data (cr)

/W :search on a word basis (default is a byte basis).

/D :search for unmatched data (default searches for matched data).

beg.addr :beginning memory search address [hex16].

end addr :ending memory search address [hex16/L byte].

data :search data [hexdata/ASCII32].

EXAMPLE:

```
>S 0,L30,12
0012
```

```
>S/W 100,110,1234
```

```
>S/D 0,10,00
```

```
0001
```

```
0002
```

```
0003
```

```
--
```

```
--
```

```
--
```

```
--
```

```
0009
```

```
000A
```

```
000B
```

```
--
```

```
--
```

```
--
```

```
--
```

```
0010
```

SUPERVISOR status - displays the current supervisor state;

●>SU (cr)

SUPERVISOR specification - sets a breakpoint as a supervisor call. When executed, data is directly transferred between the program being emulated and the ICD;

●>SU/ C|7|U ON|OFF (cr)

C :hardware break C as a supervisor call.
7 :software break 7 as a supervisor call.
U :user software break as a supervisor call.

ON :set if used as a supervisor call.
OFF :set is not used as a supervisor call.

EXAMPLE:

CHECK THE TERMINAL PORT INPUT STATE;

```
>DI 9000,900C
9000 1E01 LD E,1
9002 00 NOP
9003 B7 OR A
9004 CA0290 JP Z,9002H
9007 77 LD (HL),A
9008 04 INC B
9009 FE0D CP 0DH
900B C8 RET Z
900C 77 LD (HL),A
>SU/7 ON
>B/7 9002
>G 9000,9008
```

```
PC MC OP SP AF BC DE HL IX IY I IF(SP)
9008 04 INC B 002E FF08 1C0E 0301 A000 0000 0000 00 0 0000
<BREAK HARDWARE A>
```

TRACE status - displays the current trace mode;

●>T (cr)

TRACE designation - makes the current trace mode valid
invalid or clears the trace setting;

●>T ON|OFF|CLR (cr)

TRACE specification - sets the ICD trace mode;

● >T /S A|J , beg.addr , end addr (cr)

/S :single step mode. In this mode the All or Jump trace is executed each time the space bar or (cr) is pressed.

A :all commands are traced and displayed.

J :jump trace (only branch command is traced and displayed).

beg.addr :beginning trace address [hex16].

end addr :ending trace address [hex16/] (default=FFFF).

EXAMPLE:

SET THE TRACE MODE;

>T A,100,405

DISPLAY TRACE STATUS;

>T
(ON) ALL 0100-0300

EXECUTE TRACE EXAMPLE;

>G 100

PC	MC	OP		SP	AF	BC	DE	HL	IX	IY	I	IF(SP)
0100	2100A0	LD	HL,0A000H	0002	0023	1481	1501	A000	0000	0000	00	0 8185
0103	1100B0	LD	DE,0B000H	B000
0106	D5	PUSH	DE	0000 8000
0107	110003	LD	DE,0300H	0300
010A	7E	LD	A,(HL)	FF23
010B	12	LD	(DE),A
010C	13	INC	DE	0301
010D	C1	POP	BC	0002	B000 8185
010E	C30004	JP	0400H
0400	018016	LD	BC,1680H	1680
0403	80	ADD	A,B	1511
0404	03	INC	BC	1681
0405	F0											

RELEASE THE TRACE MODE;

>T CLR

- NOTE: TRACE DISPLAYS ">" WITH THE <ESC> KEY TO ACCEPT A COMMAND.
- NOTE: TRACE IS EXECUTED BY THE "GO" COMMAND, WHICH STARTS THE DISPLAY.

USER - allows the console terminal (connected to the ICD Terminal port) to be used as the terminal for the host system;

●>U code (cr)

code :terminator code

note: <ESC>, <NAK>, <SP>, <BS> and <cr> cannot be used as terminator codes.

EXAMPLE:

USER COMMAND WITH DESIGNATED TERMINATOR;

>U !

>A DIR B:

B: PIP

B: ED

COM : STAT

COM : ASM

COM : DUMP

COM : SYSGEN

COM : LOAD

COM : MOVCPM

VERIFY - compares the object in the Intel format (from the host computer file or specified port) with the ICD memory contents;

●>V/ T|P|A|H filename, bias (cr)

T :terminal port (ignore software handshake)
P :terminal port (perform software handshake)
A :auxiliary port
H :host port

filename :object name to compare (valid with ZICE software only).

bias :bias of file or object to compare [hexdata].

EXAMPLE:

COMPARE THE TEST.HEX FILE ON
DISKETTE WITH THE MEMORY
CONTENTS;

>V TEST
ADRS M 0
0000

>V/T, 100
ADRS M 0
0200

NOTE: "ADRS", "M" AND "O"
INDICATE THE MEMORY ADDRESS,
CONTENTS OF THE MEMORY ADDRESS
AND OBJECT, RESPECTIVELY.

ZICE command

The Zice command is used exclusively for ZICE. When the ICD is used in the remote operation (host computer) the ZICE utility software is required for controlling the ICD. ZICE expands the commands not built into the ICD (see software specification for details).

●>Z (Z command & parameter)

Z commands include;

ZB :batch processing. Executes commands of the filename. BAT file in the host system by batch processing.

ZC :comment. Inserts a comment in a batch file of the host system or in the middle of repeat processing and outputs it to the console.

ZD :directory. Displays a directory of the *.BAT or *.HEX file in the host system by directory format.

ZH :help. Displays command list.

ZM :menu. Enter data by command menu method.

ZP :print. Outputs the contents displayed on the console screen to a list device.

ZR :repeat. Repeats a specified command.

ZW :wait. Places a temporary pause in a repeat or batch subcommand.

Quick-Commands

ASSEMBLE specification:

>A mem.addr (cr)
xxxx (Z80 assemble code) (cr)
xxxx (cr)

BREAK status:

>B (cr)

Hardware BREAK specification:

>B/ A|B|C M|MR|PW|OF , addr , passcount (cr)
P|MW|PR|IA

Hardware BREAK designation:

>B/ A|B|C ON|OFF|CLR (cr)

Software BREAK specification:

>B/ 0-7 addr , passcount (cr)

Software BREAK designation:

>B/ 0-7 ON|OFF|CLR (cr)

Software BREAK op code specification:

>B S= EN|DI (cr)

External BREAK hi/low designation:

>B/X HI|LO (cr)

External BREAK designation:

>B/X ON|OFF|CLR (cr)

Event BREAK designation:

>B/E ON|OFF (cr)

Timeout BREAK designation:

>B/T ON|OFF (cr)

Write protect BREAK designation:

>B/W ON|OFF (cr)

Hardware arm/individual BREAK specification:

>B/ A|B|C ARM|IND (cr)

BREAK initialize:

>B INI (cr)

BREAK event with passcount:

>B/E passcount (cr)

CALCULATION:

>C data ± data ± data ... (cr)

COMPARE:

>CO beg.addr , end addr , comp.addr , UP|PU (cr)

DISASSEMBLE:

>DI beg.addr , end addr (cr)

DUMP:

>D/ B|W|L beg.addr , end addr (cr)

EVENT status:

>EV (cr)

EVENT specification:

>EVST= M|MR|PR|OP A=addr D=data (cr)
P|MW|PW|IA

EVENT designation:

>EV ON|OFF CLR (cr)

EXAMINE only:

>E/ W addr (cr)

EXAMINE and change:

>E/ W beg.addr = mod.data (cr)

FILL:

>F W beg.addr , end addr , data (cr)

GO:

>G beg.addr , end addr , 2nd end addr (cr)

Realtime trace status (HISTORY):

>H (cr)

Realtime trace counter reset (HISTORY):

>H CLR (cr)

Monitor trigger storage (HISTORY):

>H BM|EM|CE|ME , range (cr)

Event trigger (HISTORY):

>H EM|EE (cr)

HISTORY format display:

>H M|D , int.point , term.point (cr)

HISTORY search:

>HS,/addr/data/ IA|MR|PR|OF , int.pointer , term.pointer (cr)
MW|PW|HA

IDENTIFICATION:

>ID (cr)

INCIRCUIT status:

>I (cr)

INCIRCUIT specification:

>I 0|1|2 (cr)

LOAD:

>L/ T|P|A|H filename , bias (cr)

MAPPING status:

>MA (cr)

MAPPING specification:

>MA beg.addr , end addr = RO|RW|NO|US (cr)

MOVE:

>M beg.addr , end addr , mov.addr , UP|PU (cr)

NEXT:

>N steps (cr)

OFFSET status:

>O (cr)

OFFSET specification:

>O &1|&2|&3|&4 = addr (cr)

PIN status:

>PI (cr)

PIN specification:

>PI BUSRQ|NMI|INTR = EN|DI (cr)

PORT examination:

>P beg.addr (cr)

PORT examination and change:

>P beg.addr = data (cr)

PRINT:

>PR ON|OFF (cr)

QUIT:

>Q (cr)

REGISTER:

>R (cr)

REGISTER reset:

>R (push)RESET (cr)

REGISTER change:

```
>R reg = data (cr)
(reg>>> A F B C D E H L I
        A'F'B'C'D'E'H'L'I'
        BC DH HL IX IY SP
        BC'DH'HL'
        CY S Z HC P N IF(IFFI)
```

SAVE:

```
>SA/ T|P|A|H filename , beg.addr , end addr , user addr (cr)
```

SEARCH:

```
>S/ W / D beg.addr , end addr , data (cr)
```

SUPERVISOR status:

```
>SU (cr)
```

SUPERVISOR specification:

```
>SU/ C|7|U ON|OFF (cr)
```

TRACE status:

```
>T (cr)
```

TRACE specification:

```
>T /S A|J, beg.addr , end addr (cr)
```

TRACE designation:

```
>T ON|OFF|CLR (cr)
```

USER:

```
>U code
```

VERIFY:

```
>V/ T|P|A|H filename , bias (cr)
```

Zice command:

```
>Z B|C|D|H|M|P|R|W (Z command & parameter) (cr)
```

THIS PAGE INTENTIONALLY BLANK

SECTION IV

TECHNICAL REFERENCES

INTRODUCTION

This section contains specific technical information on the ICD-278 and is included to allow the user to become familiar, in detail, with specific emulator operations, techniques and procedures. The areas covered in this section relate to; A) Emulation Select switch, B) Data Bus Emulation connector, C) Serial Interface, D) Memory Emulation, E) CPU Emulation, F) Probes, G) Control Modules.

A comprehensive listing of the specific headings for each chapter may be found in the Table of Contents located at the beginning of this manual.

Section IV Technical References

- A** Emulation Select switch
Describes the various settings and functions of the 4-bit emulation switch.
- B** Data Bus Emulation connector
Describes the function and use of the Data Bus Emulation connector.
- C** Serial Interface
Describes the components and functions of the S-791 Serial Interface module. Includes the procedure to connect the ICD-278 to a console terminal or host computer.
- D** Memory Emulation
Describes the relationship between the ICD and Target System with regard to memory transfer and mapping.
- E** CPU Emulation
Highlights differences between the Z80 micro-processor and the emulator. Also describes prerequisite conditions for integrating the ICD-278 with a target system.
- F** Probes
Describes the functions of the various plug probes used with the ICD-278.
- G** Control Modules
Describes and illustrates the four Control Modules within the ICD-278 and details their removal and installation.

THIS PAGE INTENTIONALLY BLANK

A EMULATION SELECT SWITCH

The Emulation Select switch allows the user to implement specialized conditions for the ICD-278 by modifying the machine cycle operation to the target system. The information contained here shows the various settings and functions of this switch (Figure 4-A.1) including the factory setting (Figure 4-A.2).

The Emulation Select switch is a 6-bit, ON/OFF type switch that is located on the CPA/CPB In-circuit connector side of the ICD-278. To change the bit settings, insert a small, pointed tool and set accordingly.

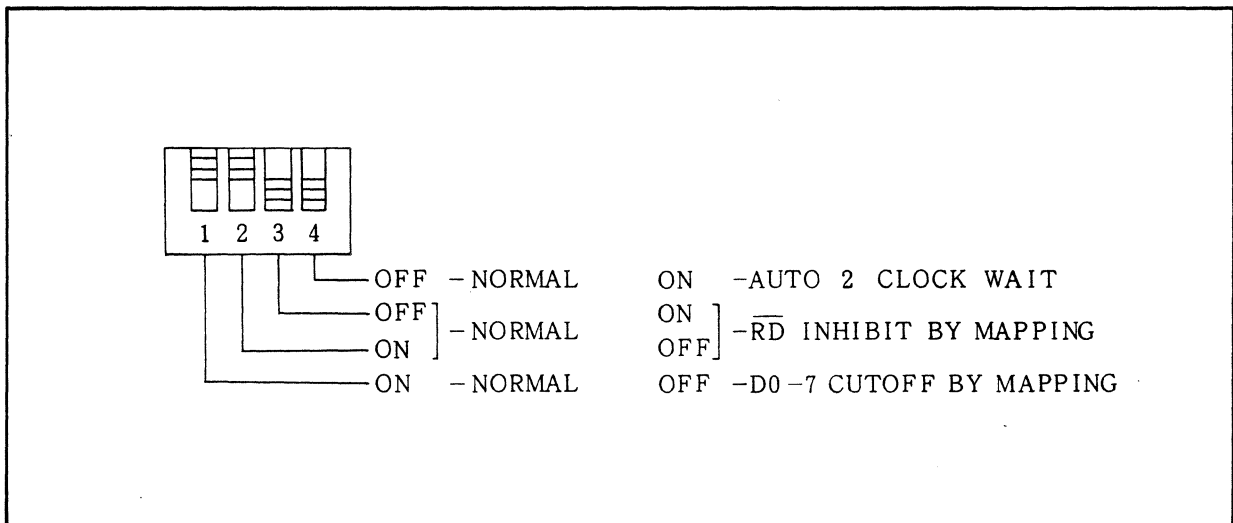


Figure 4-A.1. Emulation Select Switch

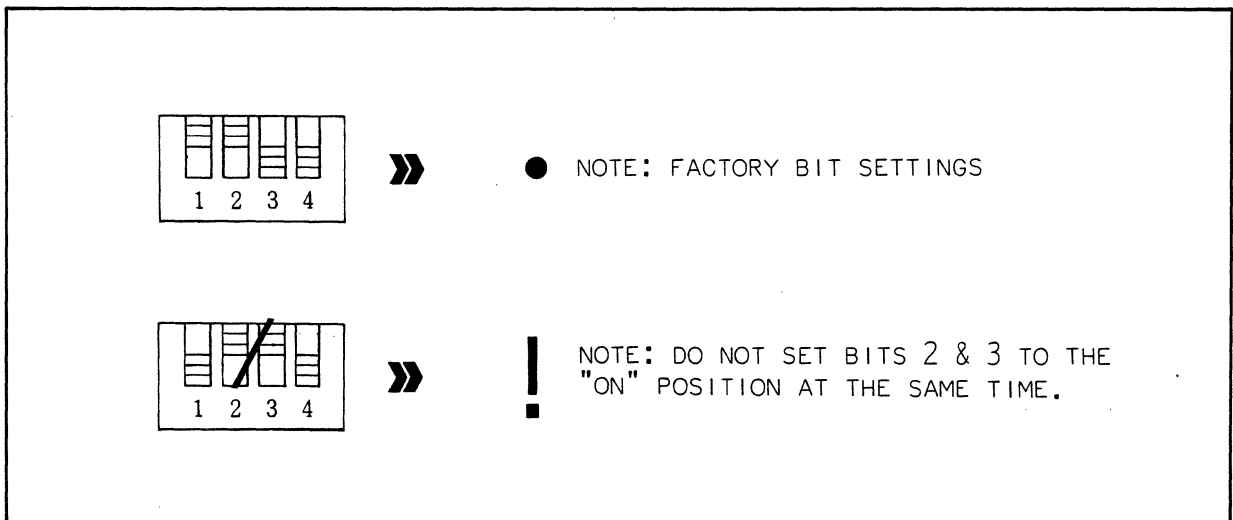
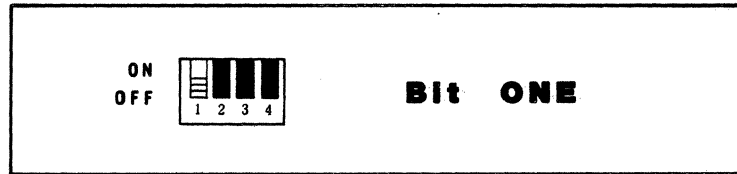


Figure 4-A.2. Factory Setting



- OFF - Disables emulator data bus D0-D7 from the target system data bus.
- ON - Normal setting. D0-D7 outputs to target system from the emulator data bus.

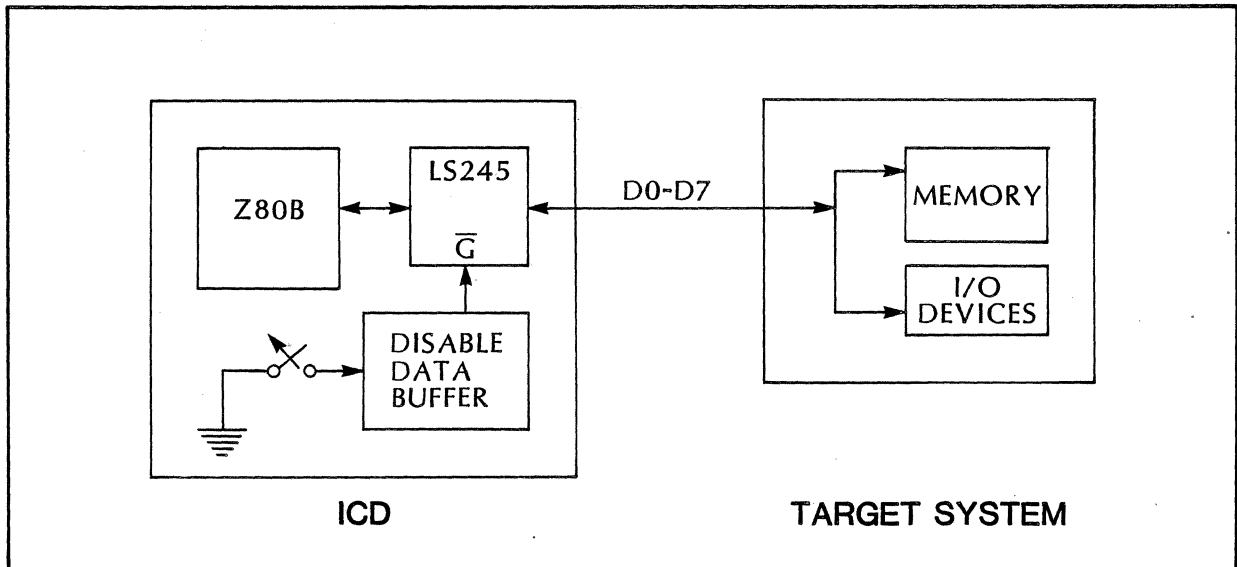
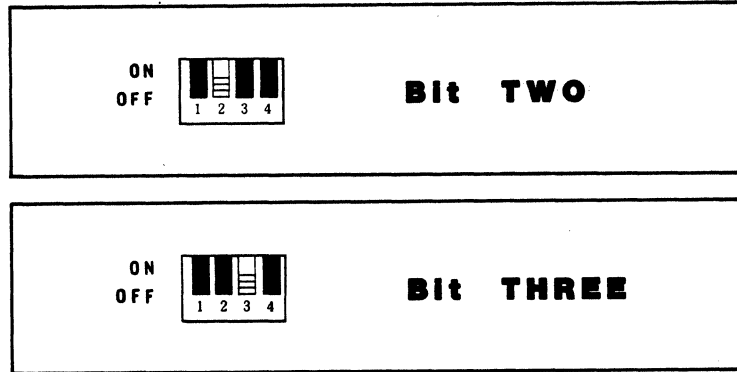


Figure 4-A.3. Emulation Data Bus Block Diagram



2 ON - RD signal outputs to the target system independently
 3 OFF of the Mapping command.

2 OFF - Use in the I1 mode only. Map command dependent.
 3 ON

Example: >I1 (cr)
 >MA 0,0FFF=RW [ICD memory]
 >MA 1000,FFFF=US [target memory]

RD signal does not output to the target system when executing out of emulator memory (RD signal outputs to the target system when executing from the target memory).

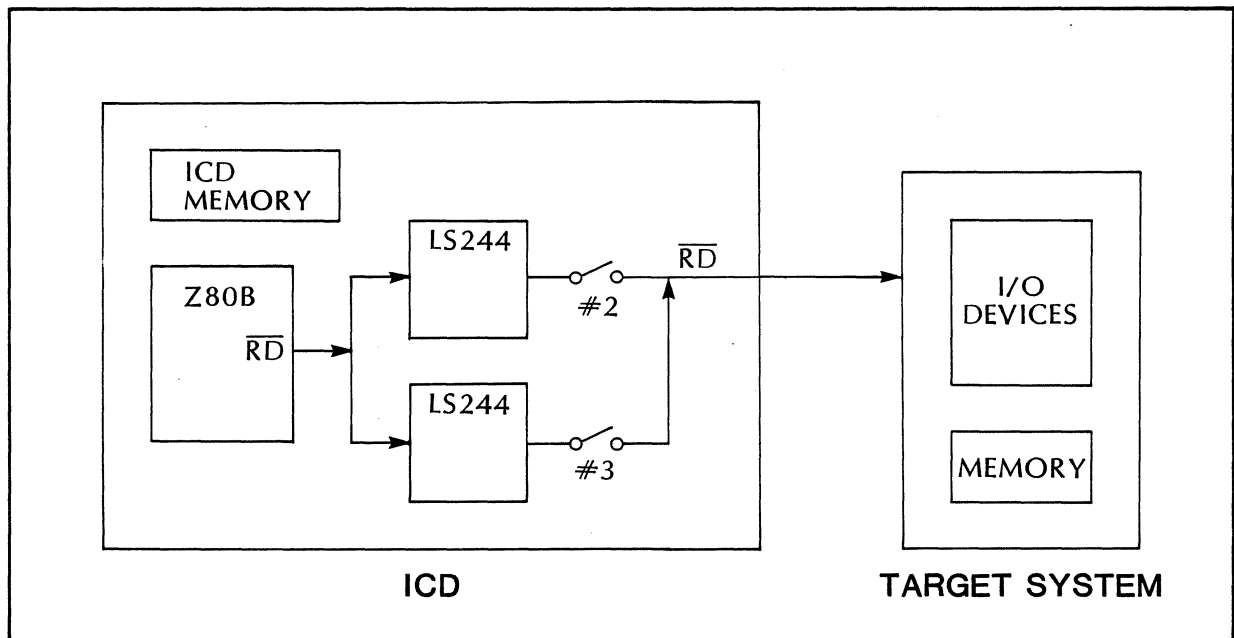
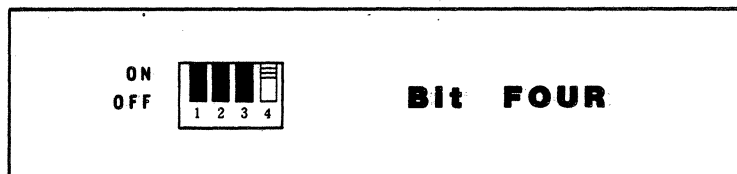


Figure 4-A.4. Emulator READ Signal Block Diagram



- ON - 1, 2 or 3 clock wait is inserted in each machine cycle.
- OFF - No clock wait is inserted in machine cycle.

The wait state produced by the ICD-278 can hold for a period of two (optional one or three) clocks (wait states) by connecting the WT, 1C and 2C points on the S-793 CPU module.

Setting the wait state;

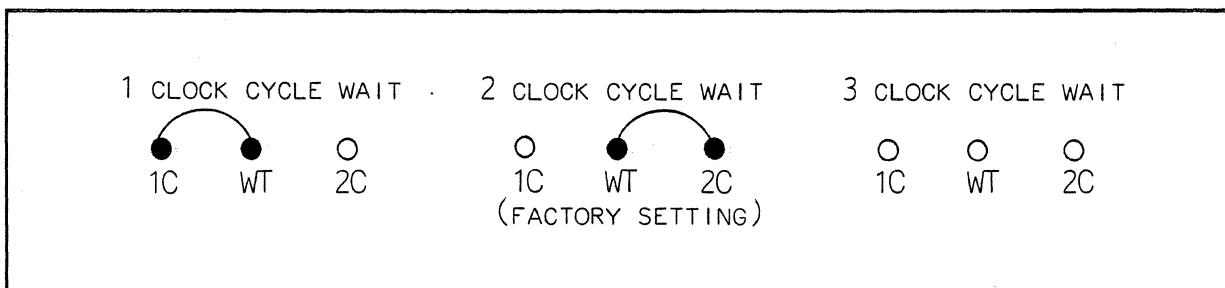


Figure 4-A.5. Wait States

B DATA BUS EMULATION CONNECTOR

The DB (Data bus) Emulation connector is used to forcibly output the "RETI" (return from interrupt) instruction to I/O devices on the target system (used in the I1 mode only). This connector is used when executing the RETI instruction out of emulator memory (absence of MREQ from data bus direction control).

Connect the DB Emulation connector directly (in-circuit) to the outside of the data bus buffer (memory and I/O side) of the target system.

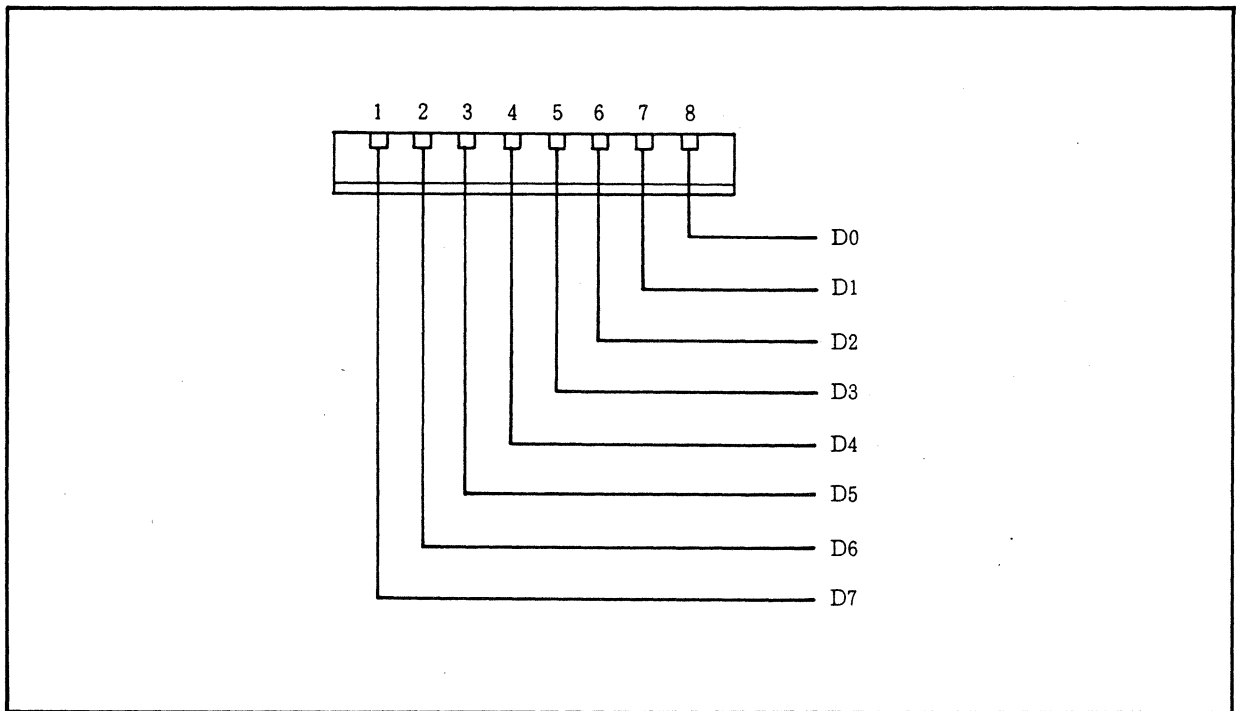


Figure 4-B.1. DB. EMUL Connector

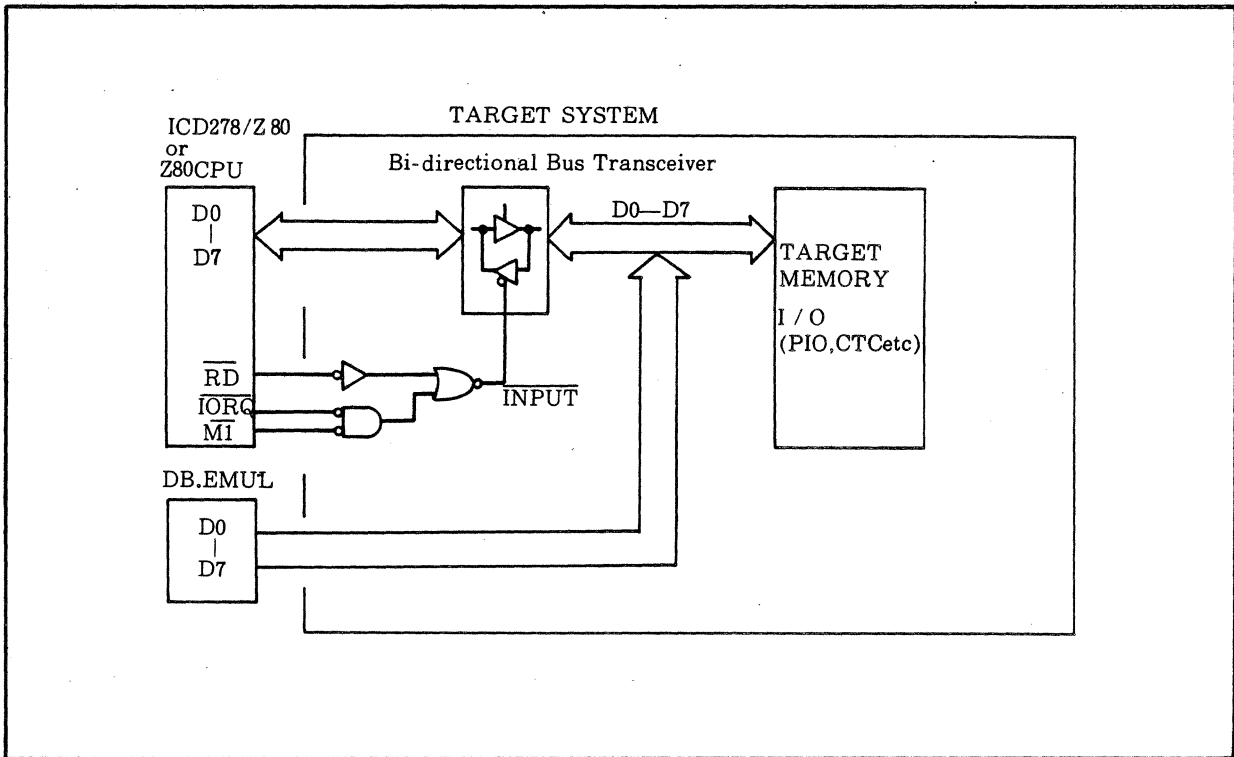


Figure 4-B.2. DB. EMUL Connector Configuration

C SERIAL INTERFACE

- ① Option jumper socket - JA is an optional jumper socket to connect the interface signal of the Terminal port to the current loop and TTL.
- ② JB is an optional jumper socket to connect the interface signal of the Host/Aux port to the current loop and TTL. (RS-232C interface is standard)
- ③ Terminal and Host/Aux line drivers - SN75188 is mounted in the RS-232C and current loop interface. For TTL interface, JA3,4 & 5 or JB3,4 & 5 must be changed for replacement with SN7438.
- ④ Terminal line driver
- ⑤ Host line driver
- ⑥ JA3,4 & 5 power supply jumpers - used for Terminal line driver IC. JA3 and 5 supplies +12V to SN75188. JA4 supplies +5V to SN7438.
- ⑦ JB3,4 & 5 power supply jumpers - used for Host/Aux line driver IC. JB3 and 5 supplies +12V to SN75188. JB4 supplies +5V to SN7438.
- ⑧,9 Transmission format switch - used to set data and stop bits for Terminal and Host/Aux ports (Figure 4-B.1, B.2).
- ⑧ Sets transmission format of the Terminal port
- ⑨ Sets transmission format of the Host/Aux port

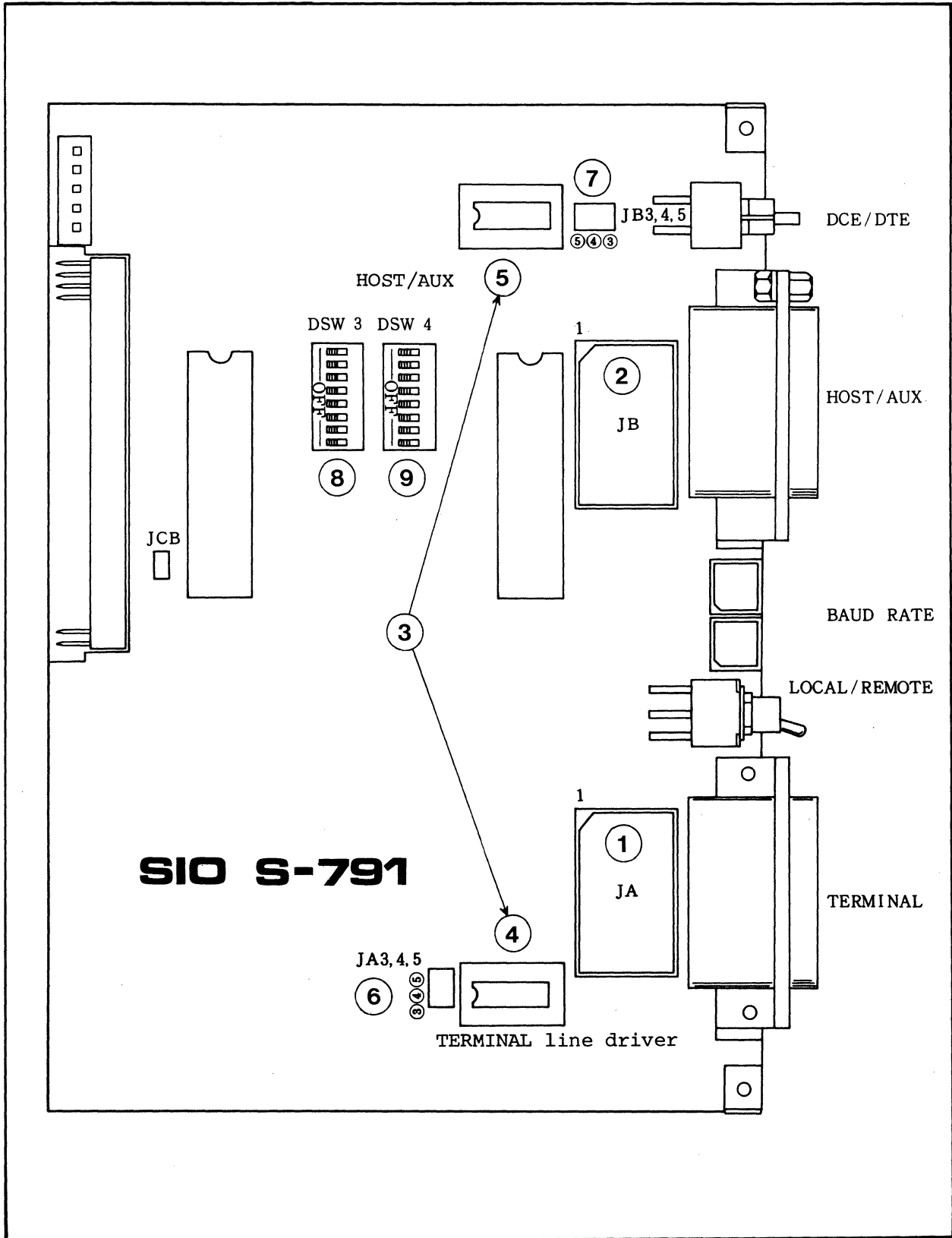


Figure 4-C.1. S-791 SIO Module

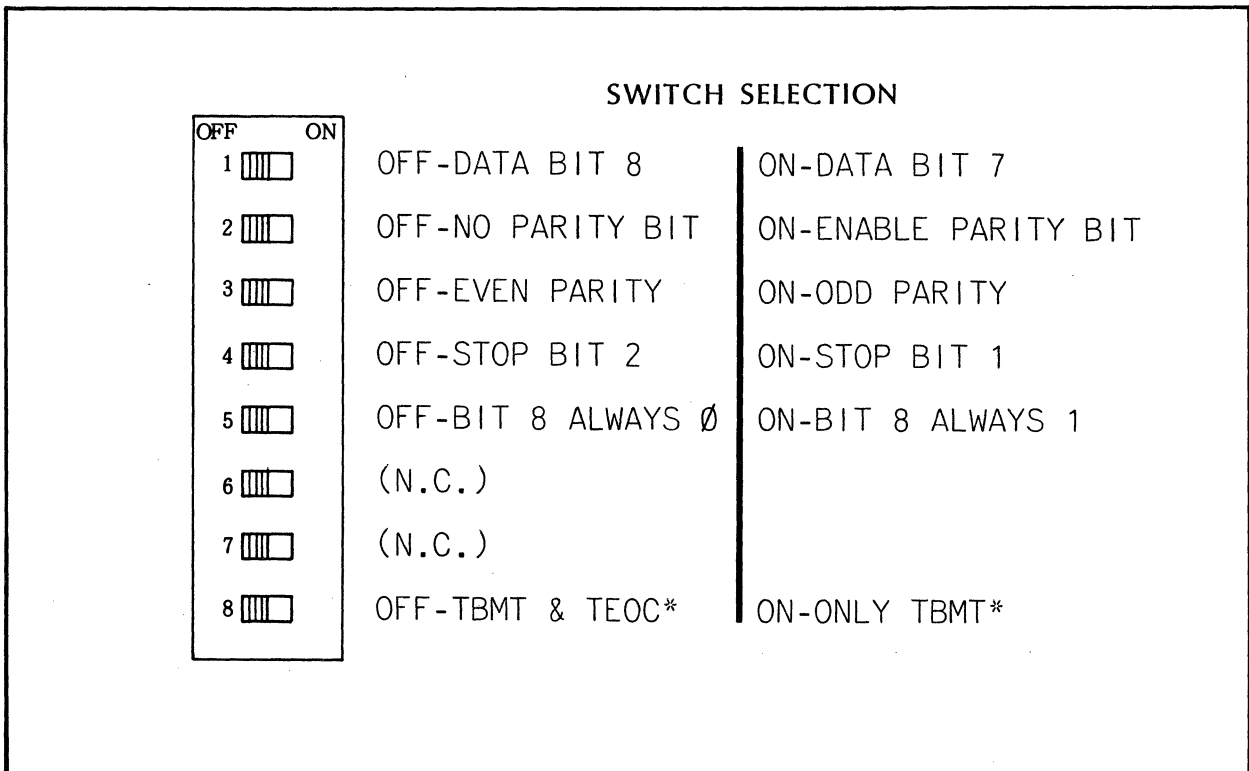


Figure 4-C.2. Transmission Format Switch

When the Transmission Format switch is set to OFF (TBMT* & TEOC*), the ICD sends data in the signal buffer and monitors the BUSY signal. When set to ON (TBMT), the ICD sends data in the double buffer and does not monitor the BUSY signal.

Factory settings;

Terminal:	8 data bits	Host/Aux:	8 data bits
	2 stop bits		2 stop bits
	no parity bit		no parity bit
	bit 8=0		bit 8=0
			TBMT* & TEOC*

* TBMT - Transmitted Buffer Empty. The transmitted buffer empty flag goes to a logic "1" when the data bits holding register may be loaded with another character.

* TEOC - Transmitted End of Character. This line goes to a logic "1" each time a full character is transmitted. It remains at this level until the start of transmission of the next character.

SERIAL INTERFACE SPECIFICATIONS

Interface channel: Two channels (Terminal & Host/Aux)

Interface signals: RS-232 C (standard specifications)
20mA current loop
TTL

The current loop and TTL can be set according to Terminal and Host/Aux by changing jumper sockets JA and JB on the S-791 board or line driver IC.

Communication method: Full-duplex, asynchronization

Transmission format: Start bit - 1
Data bits - 7 or 8 (factory; 8)
Stop bits - 1 or 2 (factory; 2)
Parity bit- Odd/Even or none
(factory; none)
The data, stop and parity bits are set by dip switches DSW3 and 4 on the S-791 board.

Transmission codes: ASCII codes

Baud Rate: 19,200 9,600 4,800 2,400
2,000 1,200 1,800 600
300 200 134.5 110 75

The baud rate is set by the Baud Rate switch on the communication panel (Figure 4-C.1).

RS-232C INTERFACE (TERMINAL)

PIN NO.	SIGNAL NAME	MEANING	IN/OUT	JA No.
1	FG	FRAME GROUND		SN 75188N ^{*3}
2	SD	SEND DATA	IN	
3	RD	RECEIVE DATA	OUT	
4	RTS	REQUEST	IN ^{*1}	J6, J20 ^{*2}
5	CTS	CLEAR TO SEND	OUT ^{*1}	
6	DSR	DATA SEND READY	OUT	
20	DTR	DATA TERMINAL READY	IN	
7	SG	SIGNAL GROUND		

Table 4-C.1. RS-232C Interface Pin Designation (Terminal)

- *1 CTS and RTS are looped back (null modem) within the ICD and pulled up to +5V.
- *2 DTR and DSR are looped back (null modem) within the ICD by connecting JA6 and 20 (pins 15 & 16) together. DTR is used as the BUSY signal for an ICD terminal by connecting JA20. DSR may be used as the BUSY signal by connecting JB6.

When JA6 is connected, the loopback jumper between JA6 and 20 (pins 15 & 16) is prohibited. The standard connection = JA6/20 and JA20.

- *3 SN75188N is installed for the TERMINAL line driver. In this configuration, JB3 and JB5 are connected with a jumper plug to supply ±12V to SN75188N. JA4 may NOT be connected. The standard connection = JB3 and JB5.

RS-232C INTERFACE (HOST/AUX)

PIN NO.	SIGNAL NAME	MEANING	IN/OUT	JA No.
1	FG	FRAME GROUND		
2	SD	SEND DATA	OUT(IN) ↓	SN 75188N *4
3	RD	RECEIVE DATA	IN(OUT)	
4	RTS *2	REQUEST TO SEND	OUT(IN)	
5	CTS	CLEAR TO SEND	IN(OUT)	
6	DSR	DATA SEND READY	IN(OUT)	J6,20 *3
20	DTR	DATA TERMINAL READY	OUT(IN)	J6,20
7	SG	SIGNAL GROUND		

Table 4-C.2. RS-232C Interface Pin Designation (Host/Aux)

*1 Values in () are assumed when the DCE/DTE select switch is set to DCE.

*2 CTS and RTS are looped back (null modem) within the ICD by connecting JB6 and 20 (pins 15 & 16) together.

*3 DTR and DSR are looped back (null modem) within the ICD by connecting JB6 and 20 (pins 15 & 16) together.

DTR is used as the BUSY signal for the ICD terminal by connecting JB20. DSR may also be used as the BUSY signal for the ICD terminal by connecting JB6. When JB6 is connected, the loopback jumper between JB6 and 20 (pins 15 & 16) is prohibited. Standard connection = JB6/20 and JB20.

*4 SN75188N is installed for the HOST/AUX line driver. In this configuration, JB3 and JB5 are connected with a jumper plug to supply +12V to SN75188N. JB4 may NOT be connected. Standard connection = JB3 and JB5.

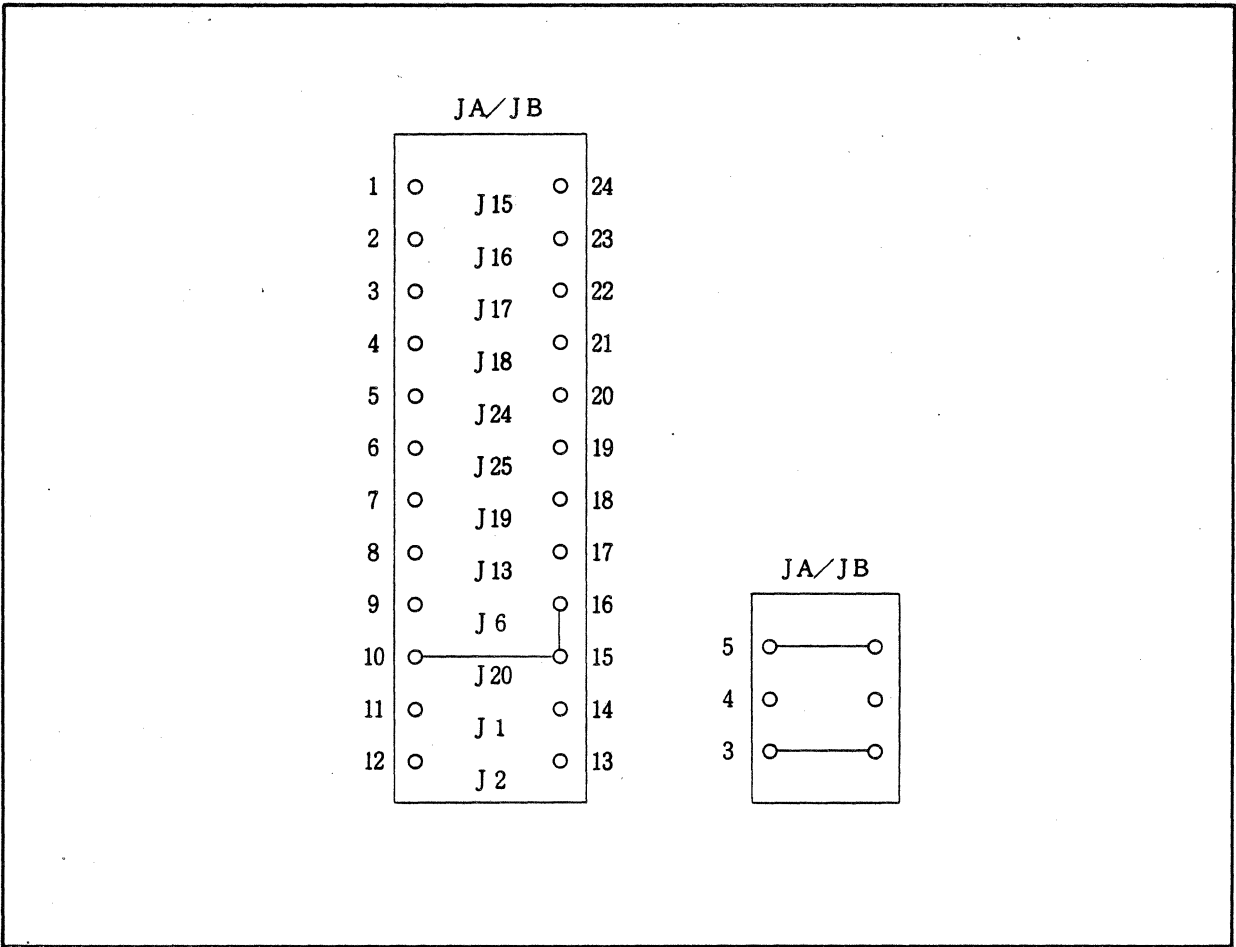


Figure 4-C.3. RS-232C Standard Jumper Settings

CURRENT LOOP INTERFACE

PIN NO.	SIGNAL NAME	MEANING	IN/OUT	JA No.
24	LOUT+	CURRENT LOOP OUT(+)	IN *1	J24
25	LOUT-	CURRENT LOOP OUT(-)	IN	J25
17	LIN+	CURRENT LOOP IN(+)	OUT *2	J17
18	LIN-	CURRENT LOOP IN(-)	OUT	J18/220 *3
15	RSTP+	READER STEP(+)	OUT *2	J15
16	RSTP-	READER STEP(-)	OUT	J16 *4

Table 4-C.3. Current Loop Interface Pin Designation (Term/Host/Aux)

- *1 Current source pin used for current loop input signals, -12V pull-down.
- *2 Current source pin used for current loop input signals, +12V pull-up.
- *3 Jumpers JA and JB18 are connected with a current limiting resistance of 220 ohms-1/4W -or- the resistance is altered to suit the associated circuit.
- *4 Jumpers JA and JB18 are connected with a current limiting resistance of 47 ohms-1/4W.

With a current loop interface, the HOST/AUX port is used only at the DCE setting (DCE/DTE select switch).

The baud rates over the current loop interface must NOT exceed 600bps.

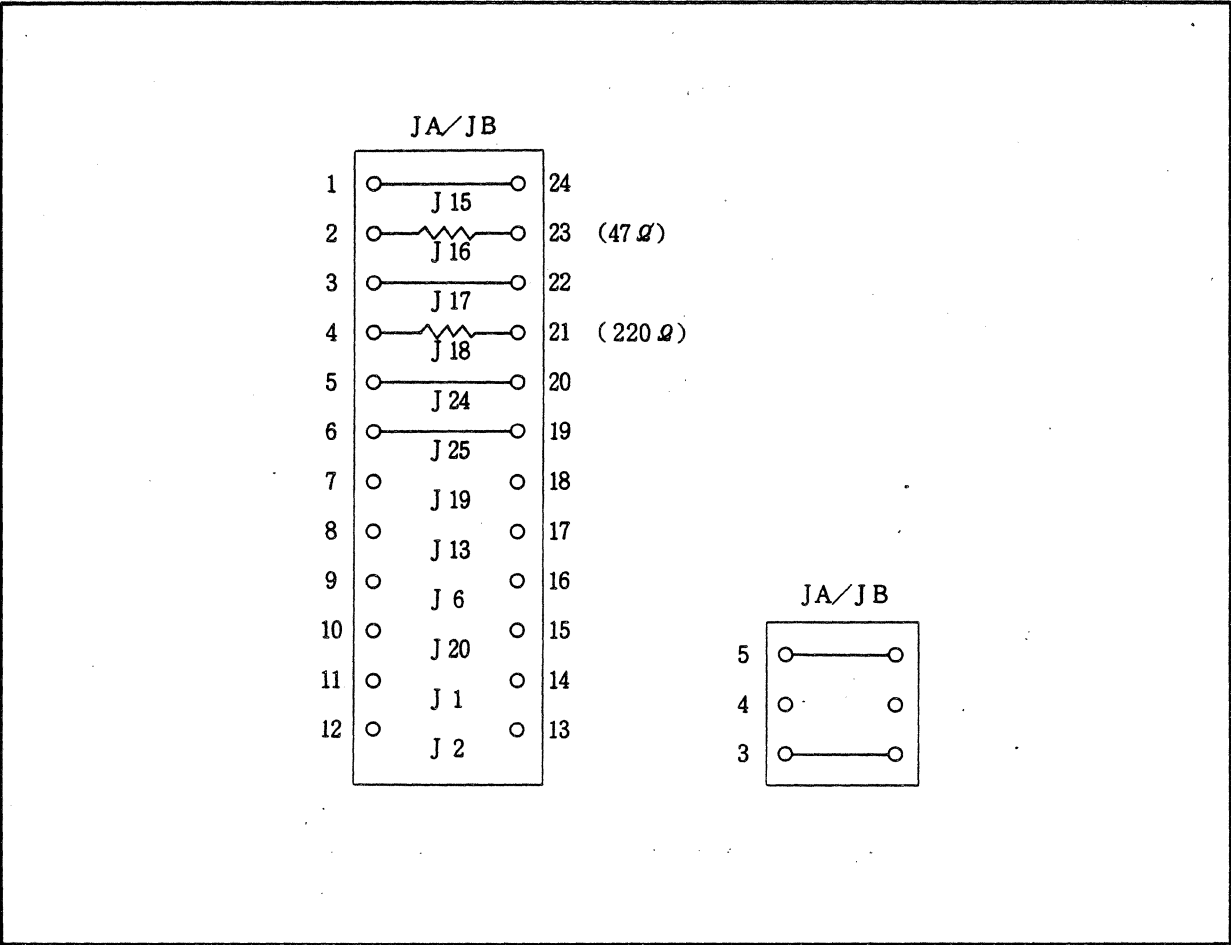


Figure 4-C.4. CLI Standard Jumper Settings

TTL INTERFACE (TERMINAL)

PIN NO.	SIGNAL NAME	MEANING	IN/OUT	JA No.
1	FG	FRAME GROUND		
2	SD	SEND DATA	IN	SN 7438 ^{*2}
3	RD	RECEIVE DATA	OUT	
19	BUSY	BUSY INPUT	IN	J19
13	BUSYOUT	BUSY OUTPUT	OUT	J13, 6 ^{*1}
16	RSTP	READER STEP	OUT	J16
7	SG	SIGNAL GROUND		

Table 4-C.4. TTL Interface Pin Designation (Terminal)

- *1 The BUSYOUT signal becomes a TTL level signal when JA6 and JA13 (pins 8 & 9) are joined together and then connected to JA13.
- *2 The Terminal line driver contains an SN7438N. JA4 is connected with a jumper plug to supply +5V to SN7438N. JA3 and 5 must NOT be jumped.

CAUTION: JA20 MAY NOT BE CONNECTED WHERE JA19 IS CONNECTED.

TTL INTERFACE (HOST/AUX)

PIN NO.	SIGNAL NAME	MEANING	IN/OUT	JA No.
1	FG	FRAME GROUND		
2	SD	SEND DATA	OUT(IN) ^{*1}	SN 7438 ^{*3}
3	RD	RECEIVE DATA	IN(OUT))	
19	BUSY	BUSY INPUT	IN	J19
13	BUSYOUT	BUSY OUTPUT	OUT	J13, 6 ^{*2}
16	RSTP	READER STEP	OUT	J16
7	SG	SIGNAL GROUND		

Table 4-C.5. TTL Interface Pin Designation (Host/Aux)

- *1 When the DCE/DTE switch is set to DCE, signals are routed in the directions indicated in ().
- *2 The BUSYOUT signal becomes a TTL signal when JB6 and JB13 (pins 8 & 9) are connected, then outputs to the pin when JB13 is connected.
- *3 The HOST/AUX line driver contains SN7438N. JB4 is connected with a jumper plug to supply +5V to SN7438N. JB3 and 5 must NOT be connected together.

CAUTION: JB20 MAY NOT BE CONNECTED WHERE JB19 IS CONNECTED.

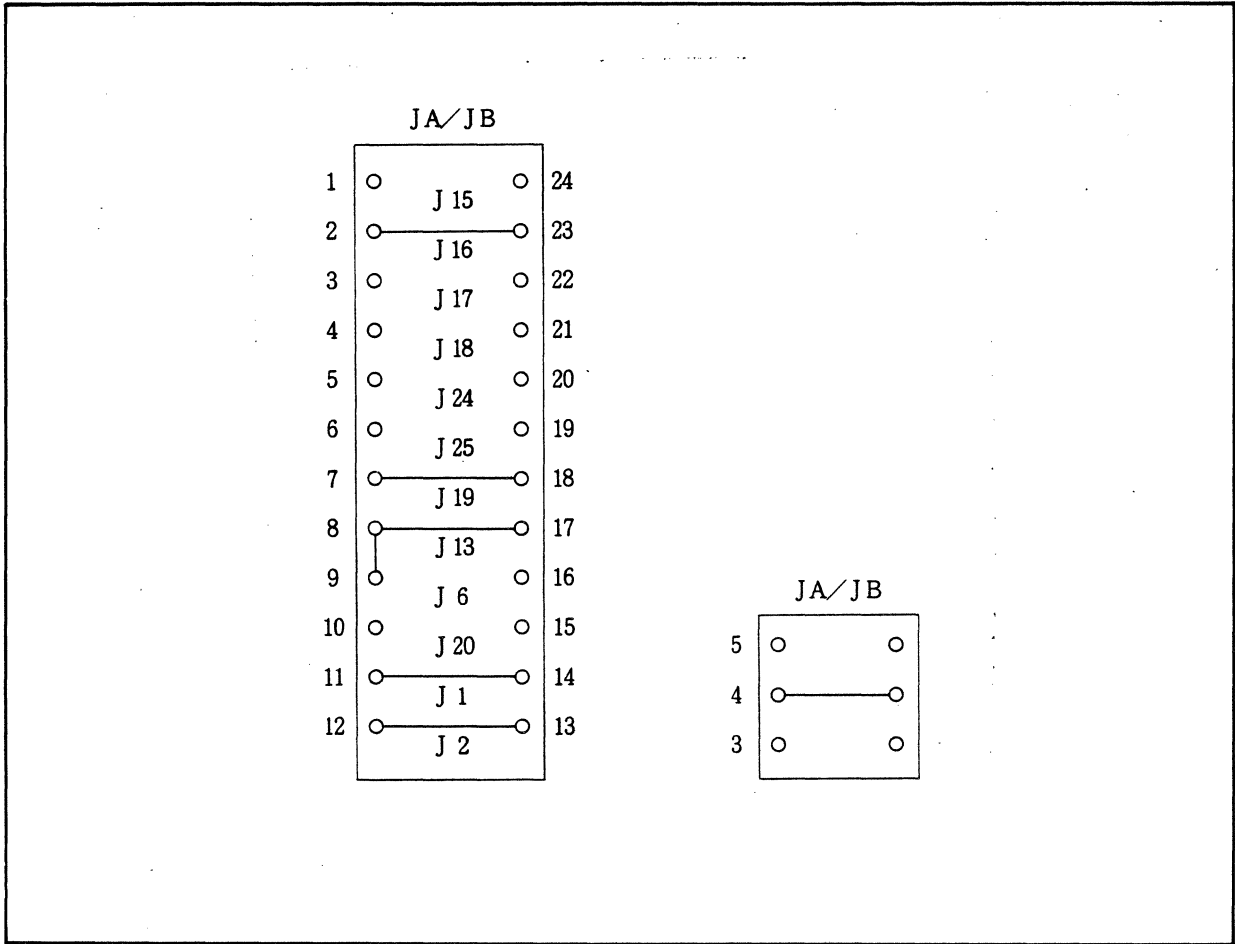


Figure 4-C.5. TTL Standard Jumper Settings

SERIAL INTERFACE CONTROL SIGNAL TIMING

XON and XOFF Protocol

When the host computer or terminal cannot receive data sent from the ICD because of a high baud rate, the data from the ICD may be controlled with XON and XOFF.

XOFF ... DC3 (CTR-S: 12H0)

XON ... DC1 (CTR-Q: 10H)

The host computer or terminal sends XOFF to the ICD before the reception buffer overruns. After sending XOFF, the host computer/terminal sends XON to the ICD if the reception buffer is ready.

The ICD stops transmitting data when it receives XOFF and ignores any incoming code other than XON thereafter. The ICD then resumes data transmission after receiving XON.

BUSY and DTR Inputs

The ICD can stop data transmission if a BUSY signal is detected from a low-speed terminal. Normally, the terminal sets the BUSY signal to L, from the leading edge of the receive data (RD) starting bit to the completion of data processing. The ICD suspends its data transmission to the terminal as long as the BUSY signal is at L (Figure 4-C.6).

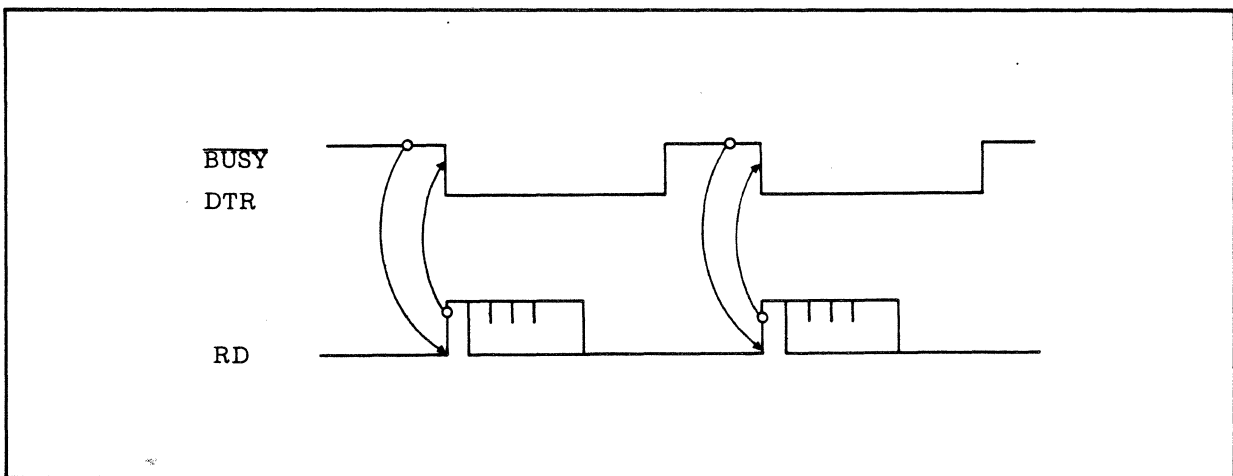


Figure 4-C.6. BUSY and DTR Input Signal Diagram

BUSYOUT and DSR Outputs

When the host computer system sends data at a higher rate of speed than that of the internal ICD monitor processor, the BUSYOUT signal of the ICD must be monitored. The ICD sets the BUSYOUT signal to L until the ICD monitor reads transmitted data (SD) from the host system. During this time, the host computer waits for data transmission to the ICD (Figure 4-C.7).

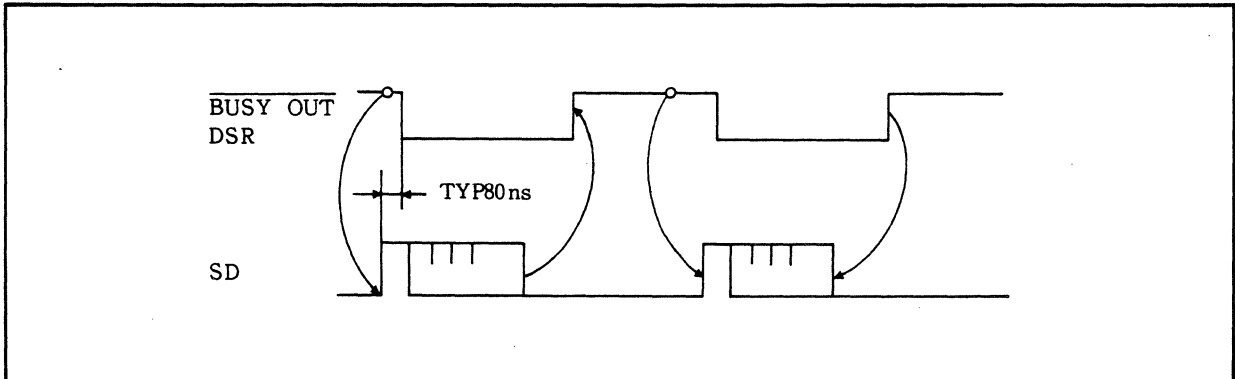


Figure 4-C.7. BUSYOUT and DSR Output Signal Diagram

RSTP Output

The ICD can transmit a RSTP signal to terminals that require a step signal on each data transmission. The ICD sets RSTP to L when data reading is requested. It then returns RSTP to H upon detection of the start bit in the transmitted data (SD) from the terminals being set (Figure 4-C.8).

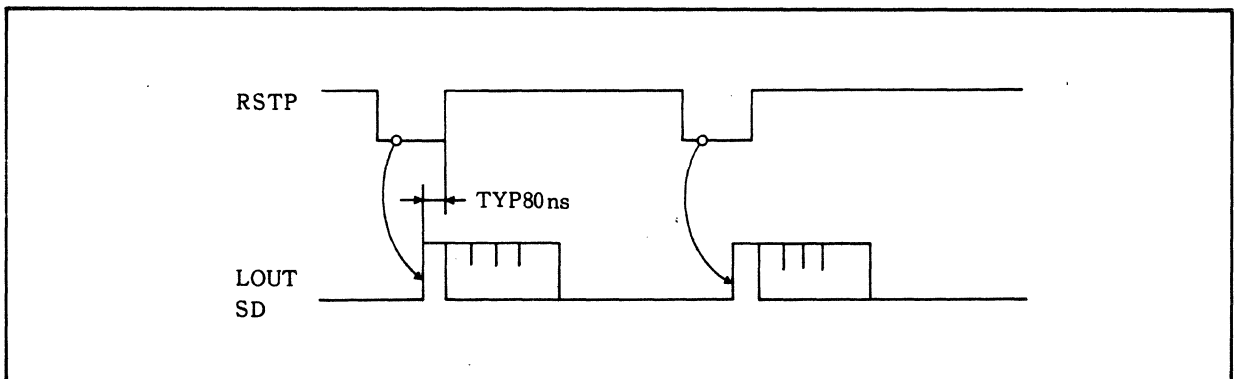


Figure 4-C.8. RSTP Output Signal Diagram

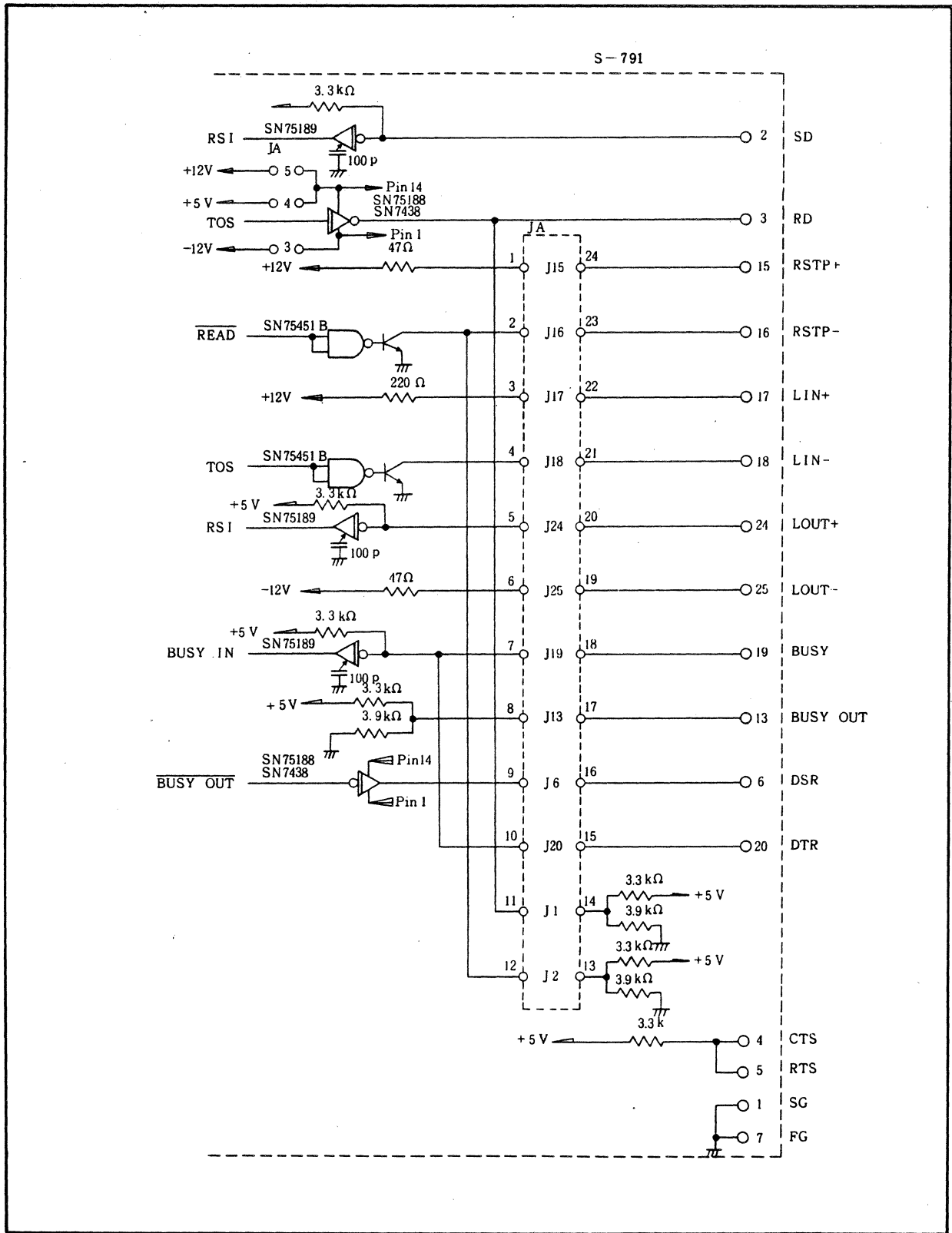


Figure 4-C.9. S-771 SIO Interface Circuit Diagram (Terminal)

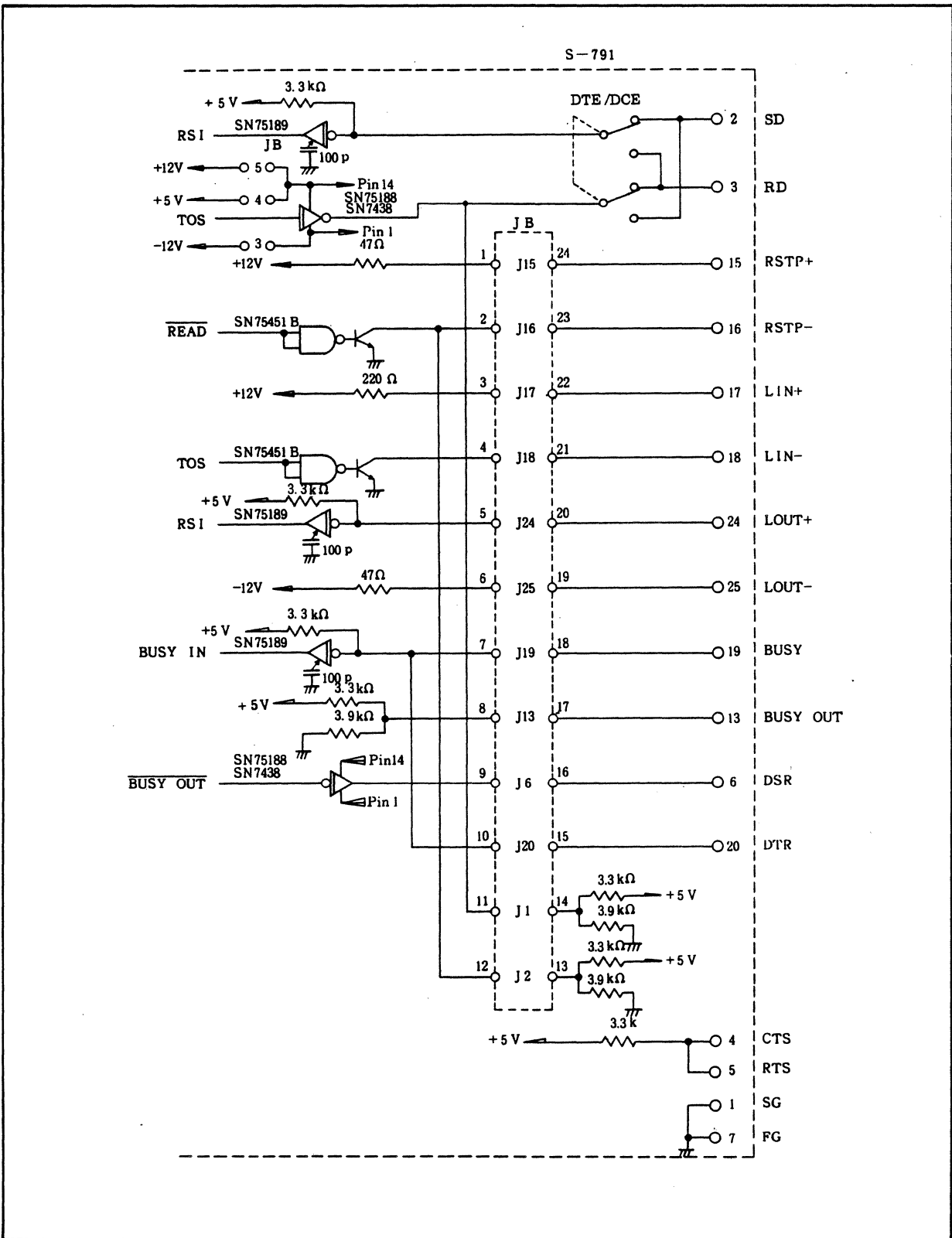


Figure 4-C.10. S-771 SIO Interface Circuit Diagram (Host/Aux)

THIS PAGE INTENTIONALLY BLANK

D ICD PROGRAM MEMORY

The ICD incorporates 64K-bytes of high speed RAM for target programs as emulation memory (Figure 4-D.1). This memory is referred to as ICD program memory (or emulation memory), in relation to user memory (target system memory).

The ICD program memory consists of high-speed static RAM, enabling support of multi-speed target systems. Also, ICD program memory differs from the usual program memory area in that it is contained within the Z80 processor, and not on a separate module. The address/data/control bus pins must assume three states during a BUSAK cycle. This prohibits DMA transfer between the target system and ICD program memory, however, DMA transfer within addressed spaces is permitted.

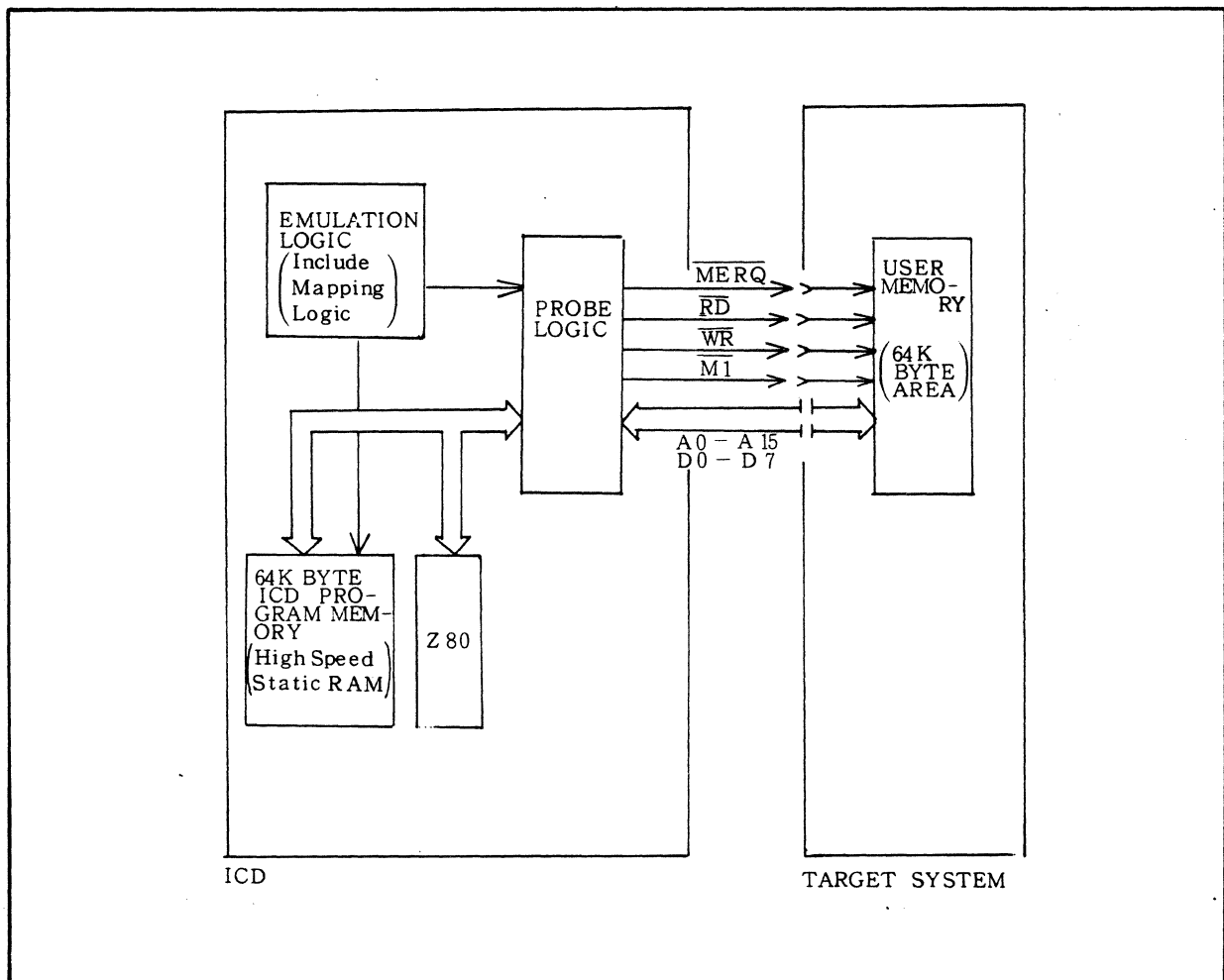


Figure 4-D.1. ICD Program Memory Diagram

USER MEMORY

User memory is memory that is installed in the target system as opposed to ICD program memory. The ICD-278 will address up to 64K bytes of target system memory (user memory).

The access time required to write to the target system memory from the ICD-278 is the same as the processors, however, the access time needed to read from the target system memory is slightly shorter than that available with the processor. Therefore, certain access time conditions must be met. These are shown in Figure 4-D.2.

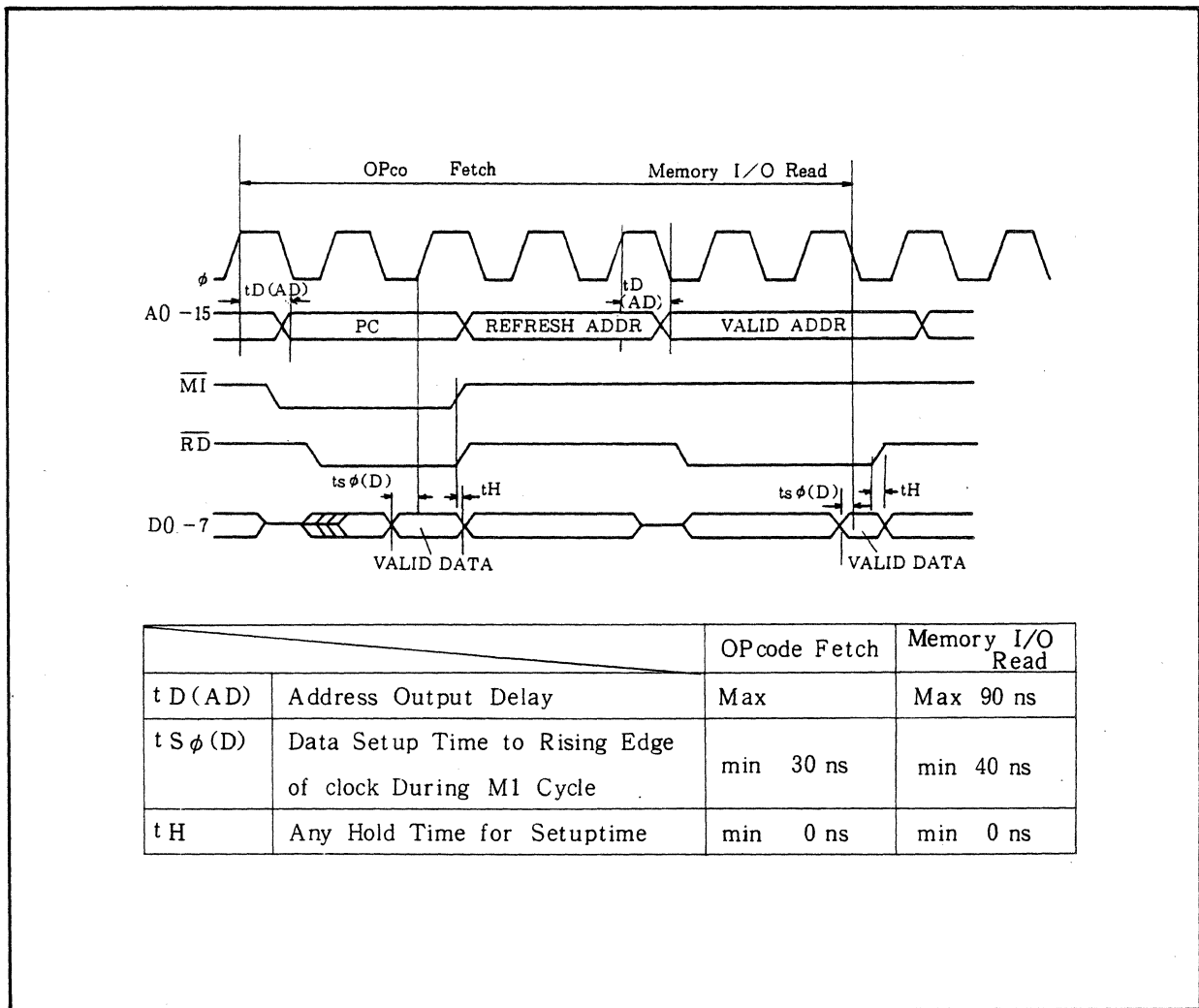


Figure 4-D.2. User Memory Timing Diagram

MEMORY MAPPING

When target system emulation is in progress during in-circuit mode I1, it is possible to specify either user memory or ICD program memory in 1K bytes (Figure 4-D.3). This specification is met using the "Map" command.

When using the ICD program memory, RW (read/write) and RO (read/only) areas can be specified in 1K-byte increments. Also, the NO (non-memory) status can also be specified on the emulation CPU memory map.

See Figure 4-D.3 for application of the following 'Memory Area' references;

- READ/WRITE : Specifies that the ICD program memory is used in place of the target system RAM. The WAIT signal from the target system is ignored when this memory area is accessed.
(R/W)
- READ ONLY : Specifies that the ICD program memory can be used in place of the target system ROM. It is NOT permitted to Write into this memory area. The "Break" command may be used to force a break if writing in this area is attempted during an emulation break. The WAIT signal from the target system is ignored when this memory area is accessed.
(RO)
- USER (US) : Specifies to use the memory mounted in the target system. The WAIT signal from the target system is ignored when this memory is accessed.
- NON (NO) : This specifies to use memory not mounted in the memory map. Read/Write is not possible with this memory during emulation. If access to this memory area is attempted, a break will occur.

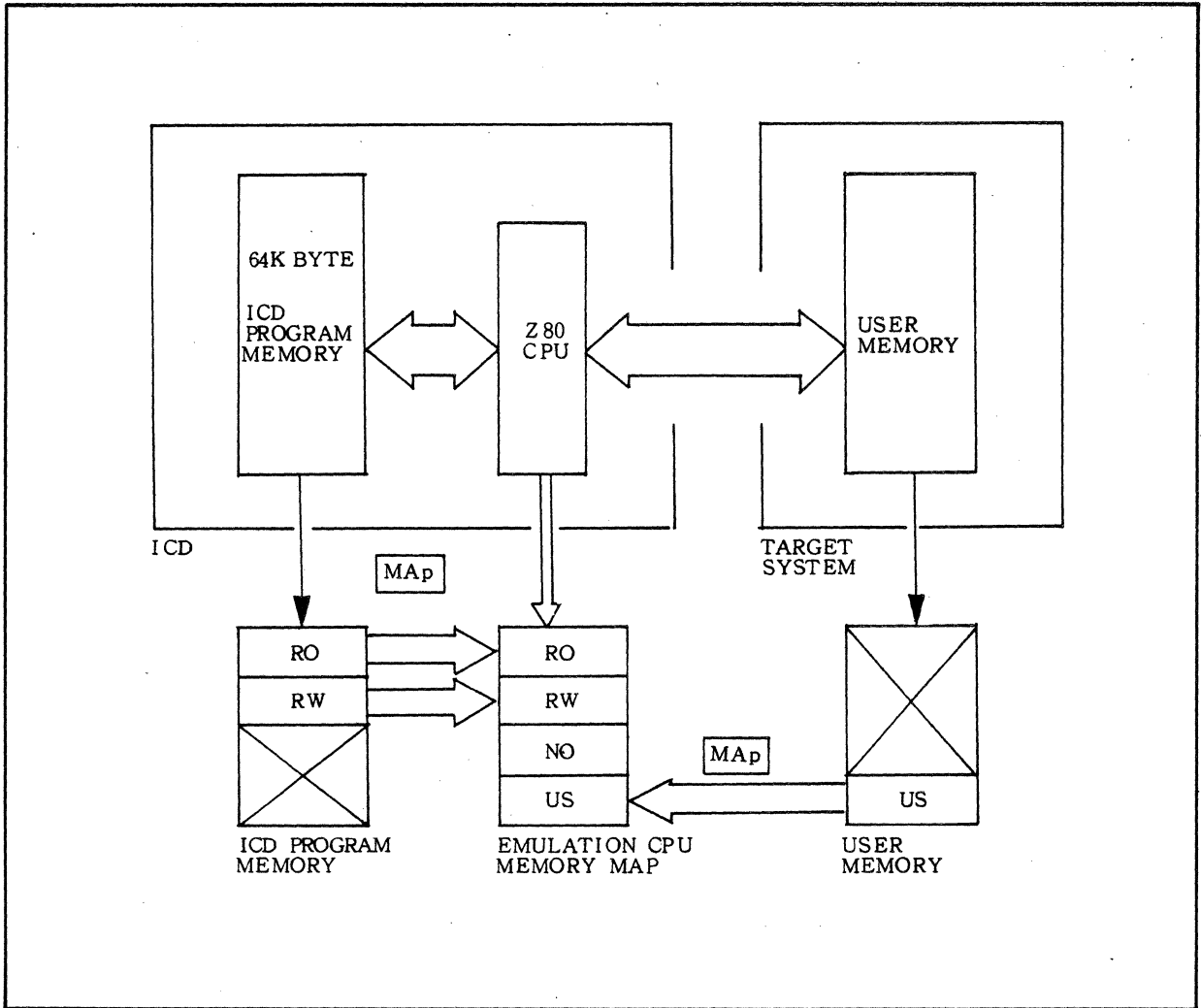


Figure 4-D.3. Mapping Configuration

E CPU EMULATION

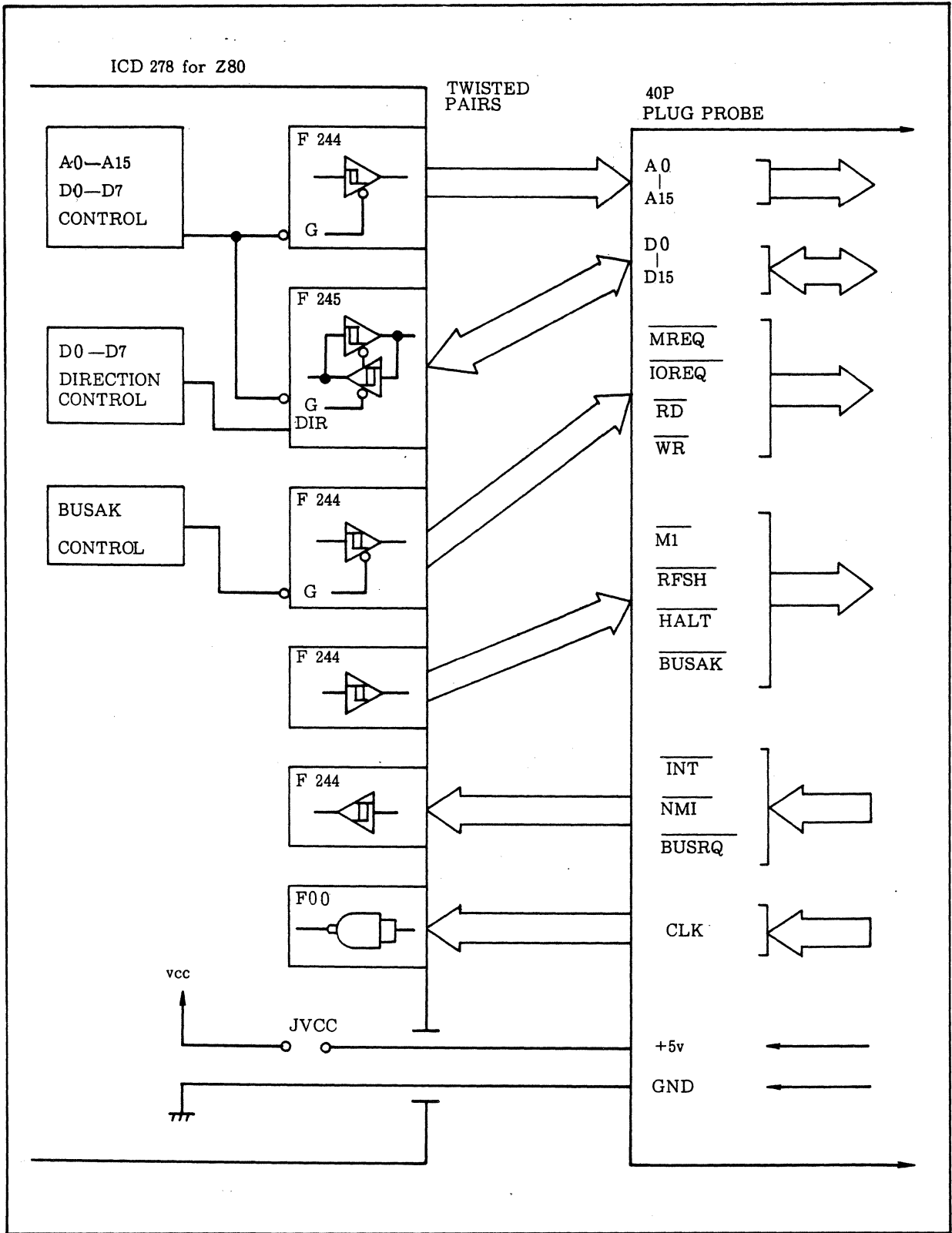


Figure 4-E.1. Emulator/Target System Interface

ICD PROGRAM MEMORY CYCLE

The ICD program memory read/write cycles do not access memory in the target system while memory mapping or during an emulation break. During the ICD program memory cycle, the MREQ-WR output signal is prohibited and therefore the memory area in the target system is unaccessible. Furthermore, the MREQ output signal which occurs during the ICD program memory write cycle is prohibited.

The data bus is specified both in the ICD program memory read and write cycles. The data then read from or written into the ICD program memory outputs to the target system.

In conventional Z80 systems, the output of the RD signal cannot be disabled because of the following:

- 1) The appearance of M1 without RD and MREQ causes the Z80 peripheral LSIs (mainly Z80 PIO) to reset.
- 2) Since the "RETI" instruction is detected by the Z80, peripheral LSIs such as PIO, CTC, SIO, DMA. Therefore, M1, RD, and D0 thru D7 signals must be present at the peripheral chip. If they are not available at the peripheral, the interrupt operation cannot be performed.

For application with any target system, the ICD uses the Emulation Select switch (or emulation method select switch) to specify the operation on the ICD program memory read cycle. Two settings are available;

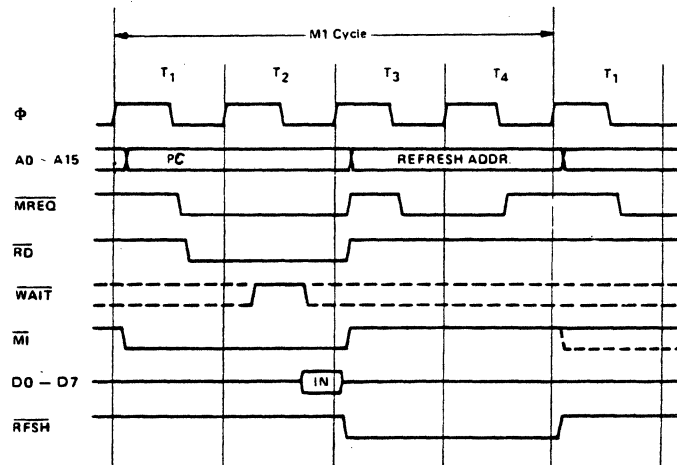
- 1) During the ICD program memory read cycle, the RD signal output is inhibited by setting the Emulation Select switch as follows:

Bit 3 = ON

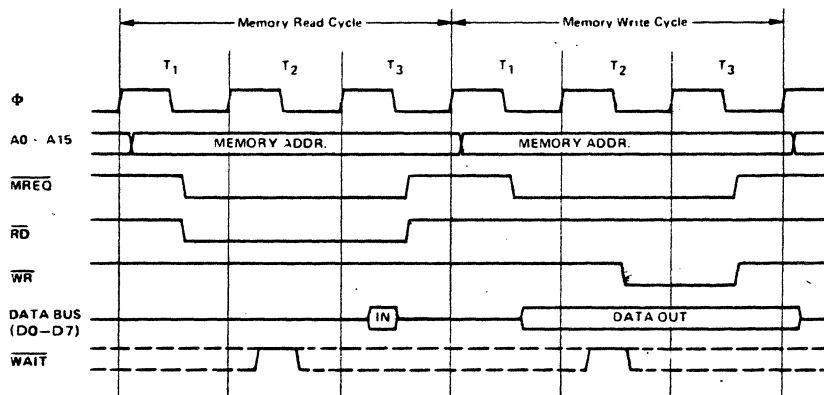
Bit 3 = OFF

- 2) During the ICD program memory read cycle, D0 thru D7 are made tristate by setting the Emulation Select switch to Bit 1 = OFF

Z80 MACHINE CYCLE

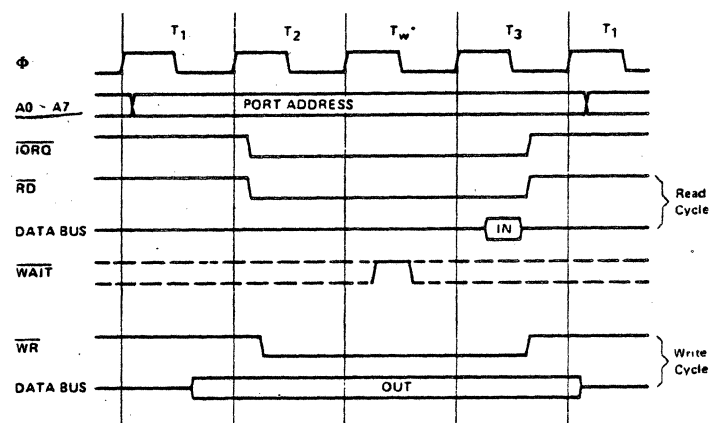


Instruction Op Code Fetch Cycle



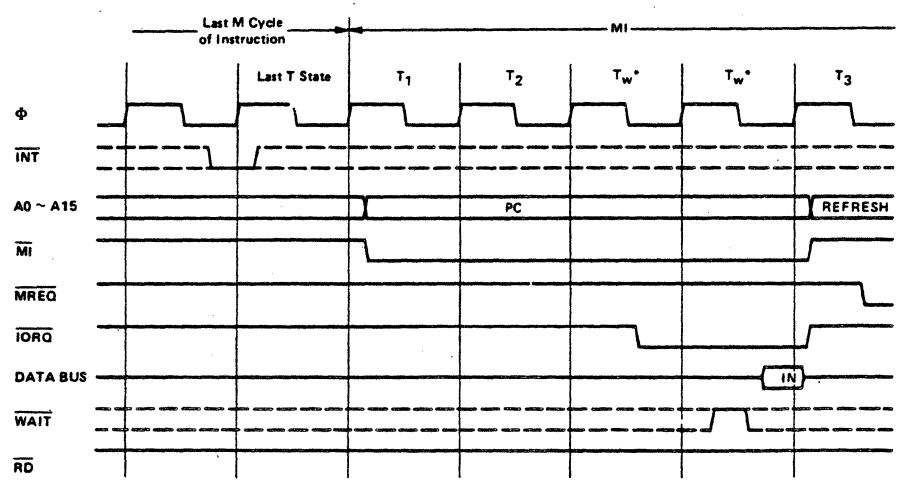
Memory Read or Write Cycle

Input or Output Cycles



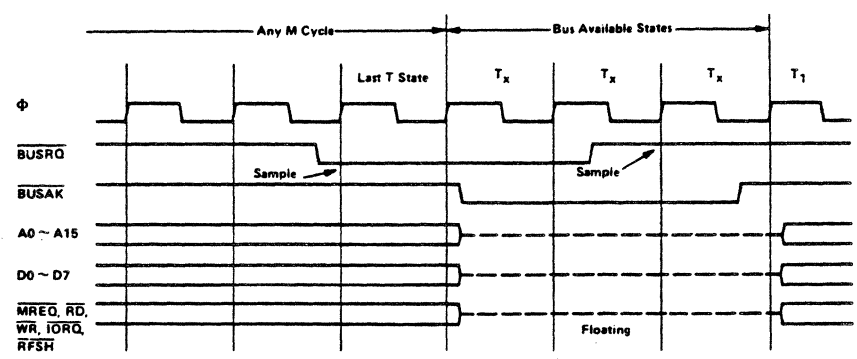
*Inserted by Z80 CPU

Interrupt Request/Acknowledge Cycle



Mode 0 shown

Bus Request/Acknowledge Cycle



CLOCK SWITCH

The CLOCK switch specifies the use of the ICD Internal Clock (4MHz) or the External Target Clock.

Internal Clock

The ICD Internal clock is specified by CLOCK=INT.

The internal clock runs at a speed of 4 MHz with a 50% duty cycle. The frequency may be changed to either 2 MHz or 4 MHz by utilizing the jumpers CX, H, and L provided on the S-793 board (Figure 4-E.3).

NOTE: The internal clock is used by the microprocessor only, it cannot be used by the target system.

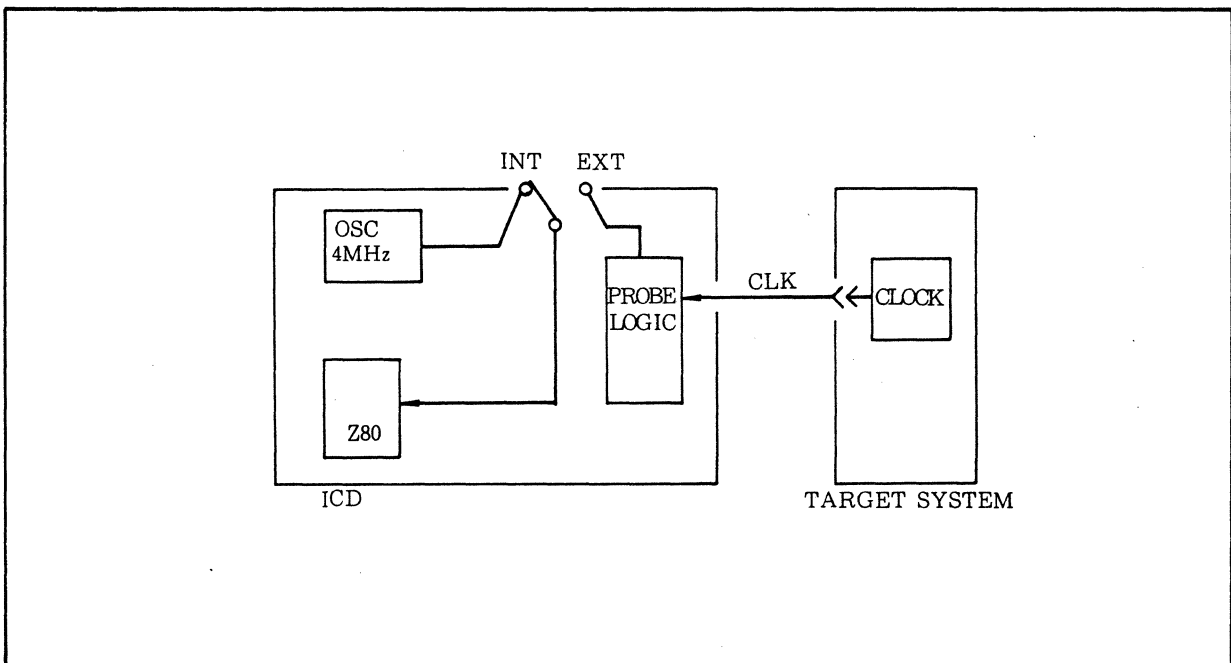


Figure 4-E.2. Clock Configuration

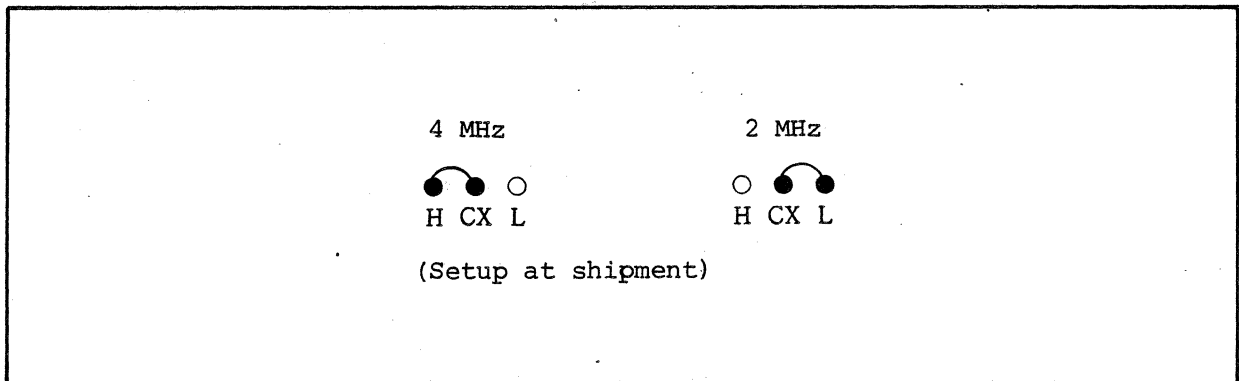


Figure 4-E.3. Setting of Internal Clock

External Clock

The External clock is specified by CLOCK=EXT.

If the external clock is used, it is possible for the target system peripheral LSI and the emulation CPU to be operated simultaneously.

NOTE: To insure accurate operation of the emulation CPU, 50% duty cycle is required for high speed clocks equal to or greater than 2.5 MHz.

The Z80 CPU requires the "H" level of the target clock to be in excess of $V_{cc}-0.6V$. If a glitch equal to or higher than 0.45V at a "L" level of the target clock appears, the M1-RD signal of the processor occurs by a glitch of the clock. If this takes place during emulation, the ICD break will have no effect. Recovery to the normal state is achieved by resetting the target system using the RESET switch.

NOTE: If this condition occurs frequently, the drive capacity of the clock driver (target system) must be improved by increasing drive levels.

RESET SWITCH

The RESET switch on the ICD Operator Panel resets the ICD only, it does NOT reset the target system. The reset switch is effective when the emulation CPU is in a break status. Target system breaks are activated using the MONITOR switch. The target system, in most cases, will have a manual reset switch that resets the entire system.

When the emulation CPU runs the target system, a manual reset of the target system causes a hardware reset of the emulation CPU or the registers in the CPU. If the emulation CPU is in an emulation break, resetting the target system will NOT cause a hardware reset of the emulation CPU. The CPU register must be reset by the "Register RES" command of the ICD monitor.

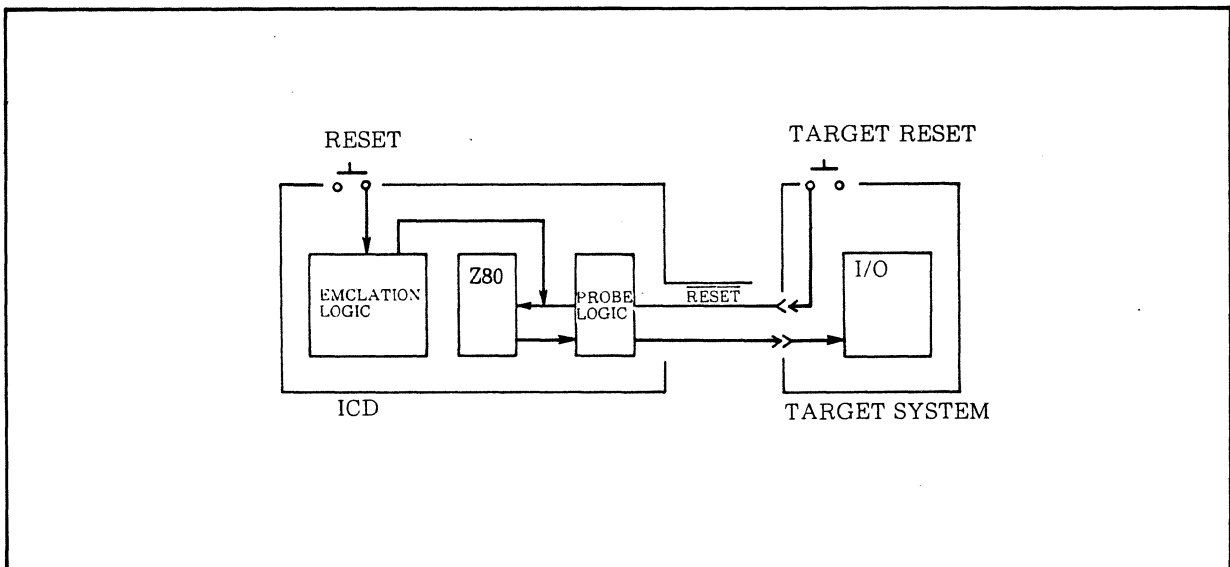


Figure 4-E.4. Reset Configuration

BUS CONTROL

The ICD can accept the BUSRQ signal any time the in-circuit mode is I1/I2. Therefore, DMA operation may be performed during an emulation break even if the target system is executing an break.

The BUSRQ signal from the target system is enabled and disabled by using the "Pin" command. This operation allows emulation regardless of the BUSRQ signal state.

The WAIT signal is effective when the target memory or I/O is accessed. The WAIT signal will generate two wait states (optional one or three) per memory cycle are automatically generated by setting the Emulation Select switch bit 4 = ON (see Technical References; A-Emulation Select switch).

This action allows the target system to operate at a higher clock speed and permits emulation even when the access time of the system or the setup time of WAIT is short.

If the WAIT signal is in a wait state for equal to or longer than 128 clocks, the access occurs as a break time-out. This condition is set using the "Break" command of the ICD monitor. The setting of a WAIT time-out introduces a break during emulation.

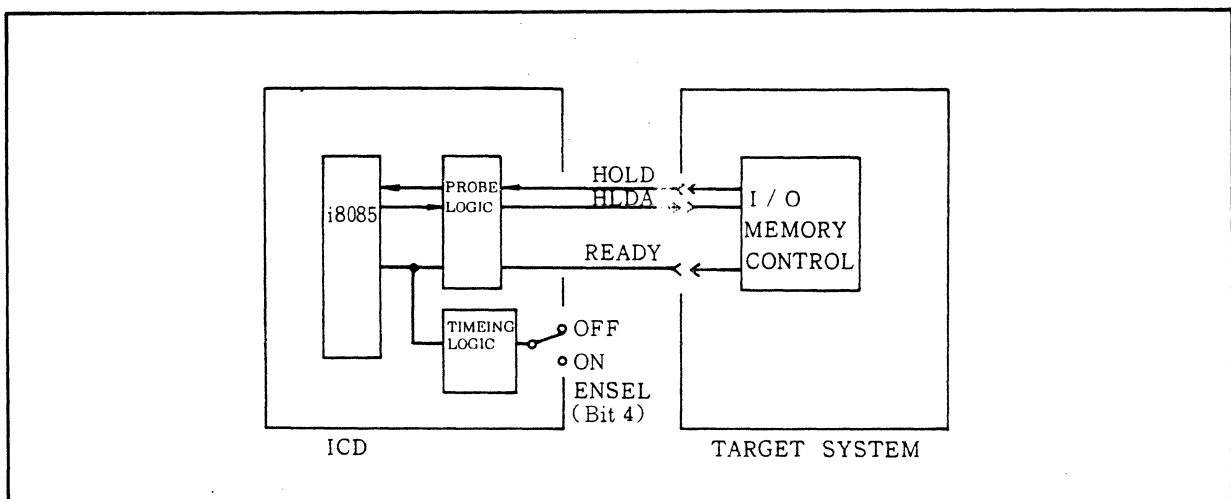


Figure 4-E.6. BUSRQ Signal Configuration

REFRESH SIGNAL

The RFSH signal outputs to the target system in any of the I1, I2 or I3 in-circuit modes. The MREQ signal for refresh is then synchronized with the RFSH signal independent of the in-circuit mode or mapping. This procedure allows refresh timing of the target system D-RAM to be synchronized with the CPU.

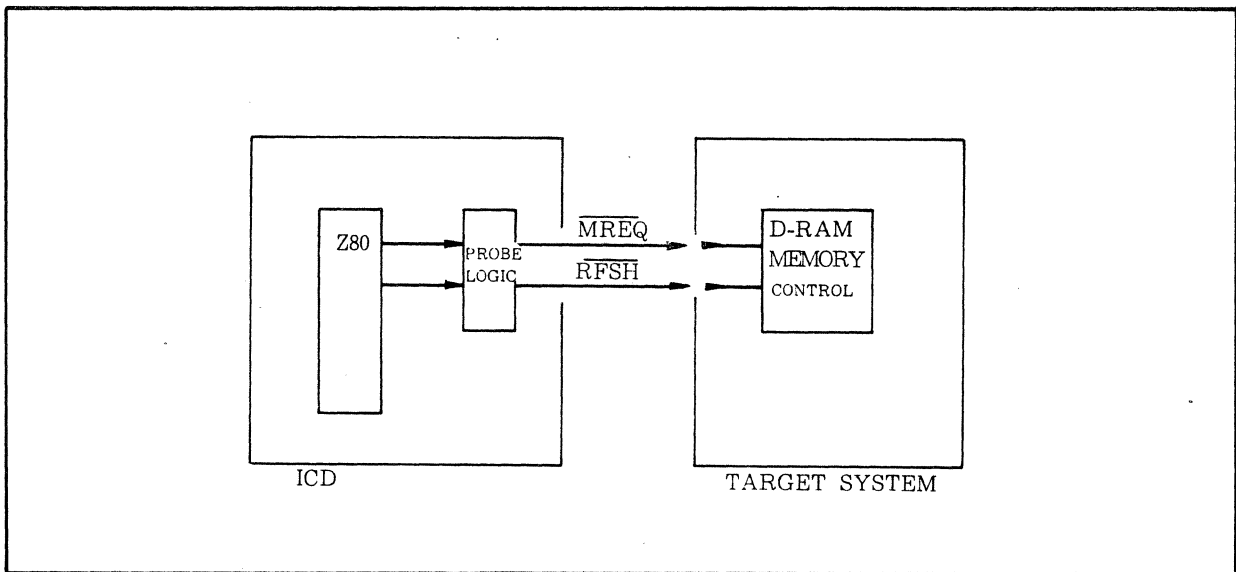


Figure 4-E.7. REFRESH Signal Configuration

Table 4-E.1. ICD-278 SIGNAL TIMING

Signal	Symbol	Parameter	Z - 80		Z - 80 A		Z 80 B		ICD 278 for Z 80		Unit
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	
ϕ	t_c	Check Period	4	30	25		165		165		μ sec
	$t_w(\phi H)$	Check Pulse Width, Clock High	180		110		65		65		ns
	$t_w(\phi L)$	Clock Pulse Width, Clock Low	180	2000	110	2000	65	2000	65	2000	ns
	t_r, f	Clock Rise and Fall Times		30		30		20		20	ns
A ₀₋₁₅	$t_d(AD)$	Address Output Delay		145		110		90		105	ns
	$t_F(AD)$	Delay to Float		110		90		80		120	ns
	t_{acm}	Address Stable Prior to \overline{MREQ} (Memory Cycle)	125		65		35		35		ns
	t_{aci}	Address Stable Prior to \overline{IORQ} (IO Cycle)	320		180		100		110		ns
D ₀₋₇	$t_d(D)$	Data Output Delay		230		150		130		145	ns
	$t_F(D)$	Delay Float During Write Cycle		90		90		80		120	ns
	$t_{S\phi}(D)$	Data Setup Time to Rising Edge of Clock During M Cycle	50		35		30		45		ns
	$t_{S\phi}(D)$	Data Setup Time to Falling Edge of Clock During M2 to M5	60		50		40		55		ns
	t_{dcm}	Data Stable Prior to \overline{WR} (Memory Cycle)	190		80		25		25		ns
	t_{dci}	Data Stable Prior to \overline{WR} (I/O Cycle)	20		10		55		55		ns
	t_{cdi}	Data Stable From \overline{WR}	120		60		30		30		ns
	t_H	Any Hold Time for Setup Time	0			0		0		0	ns
\overline{MREQ}	$t_{DL\phi}(\overline{MR})$	\overline{MREQ} Delay From Falling Edge of Clock, \overline{MREQ} Low		100		85		70		85	ns
	$t_{DH\phi}(\overline{MR})$	\overline{MREQ} Delay From Rising Edge of Clock, \overline{MREQ} High		100		85		70		85	ns
	$t_{DL\phi}(\overline{MR})$	\overline{MREQ} Delay From Falling Edge of Clock, \overline{MREQ} High		100		85		70		85	ns
	$t_w(\overline{MRL})$	Pulse Width, \overline{MREQ} Low	360		220		135		135		ns
\overline{IORQ}	$t_w(\overline{MRH})$	Pulse Width, \overline{MREQ} High	170		110		65		65		ns
	$t_{DL\phi}(\overline{IR})$	\overline{IORQ} Delay From Rising Edge of Clock, \overline{IORQ} Low		90		75		65		80	ns
	$t_{DH\phi}(\overline{IR})$	\overline{IORQ} Delay From Falling Edge of Clock, \overline{IORQ} Low		110		85		70		85	ns
	$t_{DL\phi}(\overline{IR})$	\overline{IORQ} Delay From Rising Edge of Clock, \overline{IORQ} High		100		85		70		85	ns
\overline{RD}	$t_{DH\phi}(\overline{IR})$	\overline{IORQ} Delay From Falling Edge of Clock, \overline{IORQ} High		110		85		70		85	ns
	$t_{DL\phi}(\overline{RD})$	\overline{RD} Delay From Rising Edge of Clock, \overline{RD} Low		100		85		70		85	ns
	$t_{DL\phi}(\overline{RD})$	\overline{RD} Delay From Falling Edge of Clock, \overline{RD} Low		130		95		80		95	ns
	$t_{DH\phi}(\overline{RD})$	\overline{RD} Delay From Rising Edge of Clock, \overline{RD} High		100		85		70		85	ns
\overline{WR}	$t_{DH\phi}(\overline{RD})$	\overline{RD} Delay From Falling Edge of Clock, \overline{RD} High		110		85		70		85	ns
	$t_{DL\phi}(\overline{WR})$	\overline{WR} Delay From Rising Edge of Clock, \overline{WR} Low		80		65		60		75	ns
	$t_{DL\phi}(\overline{WR})$	\overline{WR} Delay From Falling Edge of Clock, \overline{WR} Low		90		80		70		85	ns
	$t_{DH\phi}(\overline{WR})$	\overline{WR} Delay From Falling Edge of Clock, \overline{WR} High		100		80		70		85	ns
\overline{MI}	$t_w(\overline{WRL})$	Pulse Width, \overline{WR} Low	360		220		135		135		ns
	$t_{DL}(\overline{MI})$	\overline{MI} Delay From Rising Edge of Clock, \overline{MI} Low		130		100		80		95	ns
\overline{RFSH}	$t_{DH}(\overline{MI})$	\overline{MI} Delay From Rising Edge of Clock, \overline{MI} High		130		100		80		95	ns
	$t_{DL}(\overline{RFSH})$	\overline{RFSH} Delay From Rising Edge of Clock, \overline{RFSH} Low		180		130		110		125	ns
\overline{WAIT}	$t_{DH}(\overline{RFSH})$	\overline{RFSH} Delay From Rising Edge of Clock, \overline{RFSH} High		150		120		100		115	ns
	$t_s(\overline{WT})$	\overline{WAIT} Setup Time to Falling Edge of Clock	70		70		60		80		ns
\overline{HALT}	$t_d(\overline{HT})$	\overline{HALT} Delay Time From Falling Edge of Clock		300		300		260		275	ns
\overline{INT}	$t_s(\overline{IT})$	\overline{INT} Setup Time to Rising Edge of Clock	80		80		70		100		ns
\overline{NMI}	$t_w(\overline{NML})$	Pulse Width, \overline{NMI} Low	80		80		70		30		ns
\overline{BUSRQ}	$t_s(\overline{BQ})$	\overline{BUSRQ} Setup Time to Rising Edge of Clock	80		50		50		65		ns
\overline{BUSAK}	$t_{DL}(\overline{BA})$	\overline{BUSAK} Delay From Rising Edge of Clock, \overline{BUSAK} Low		120		100		90		105	ns
	$t_{DH}(\overline{BA})$	\overline{BUSAK} Delay From Falling Edge of Clock, \overline{BUSAK} High		110		100		90		105	ns
\overline{RESET}	$t_s(\overline{RS})$	\overline{RESET} Setup Time to Rising Edge of Clock	90		60		60		75		ns
	$t_F(C)$	Delay to Float (\overline{MREQ} , \overline{IORQ} , \overline{RD} and \overline{WR})		100		80		70		120	ns
	t_{mt}	\overline{MI} Stable Prior to \overline{IORQ} (Interrupt Ack.)	920		565		365		365		ns

THIS PAGE INTENTIONALLY BLANK

F PROBES

EVENT TRIGGER PROBE

The Event trigger probe detects a LOW pulse (TTL) signal when an event point is passed during the emulation process.

The duration of the pulse is 250ns (minimum) from the moment at which the RD/WR rises in a cycle with matching event conditions - to the point at which the RD/WR signal in the next cycle rises. A LOW pulse synchronized with the RD/WR signal outputs if "Don't care" has been specified as an event condition.

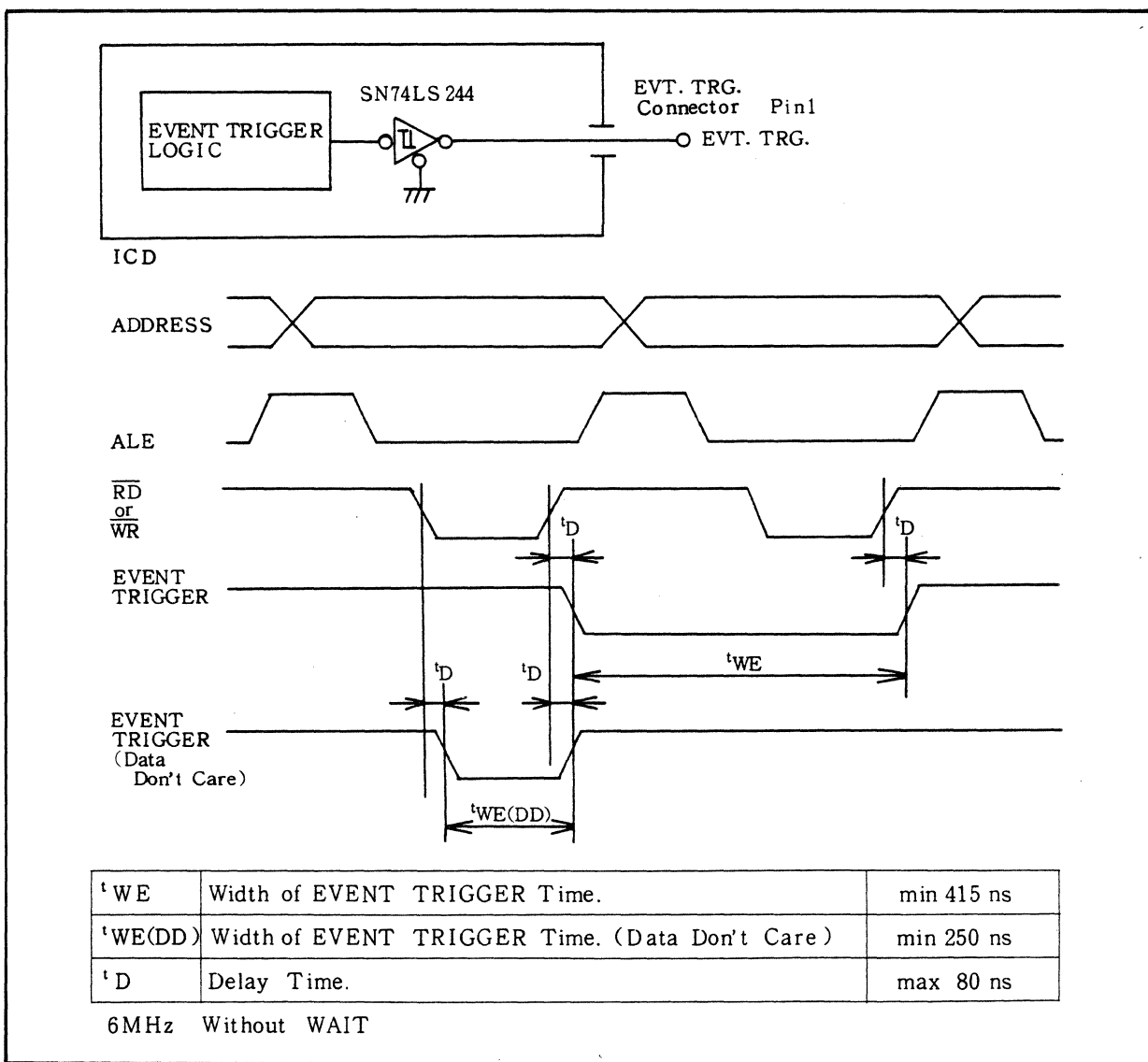


Figure 4-F.1. Event Trigger Output Timing Diagram

EMULATION QUALIFY OUTPUT

The Emulation Qualify probe determines the status of ICD emulation. The HIGH level (TTL) signal is sent from the point at which the target program is executed (with the "GO" command) to the breakpoint. Unwanted signals can be cut by connecting the Emulation Qualify output to the enable or qualify input of a logic analyzer or similar device.

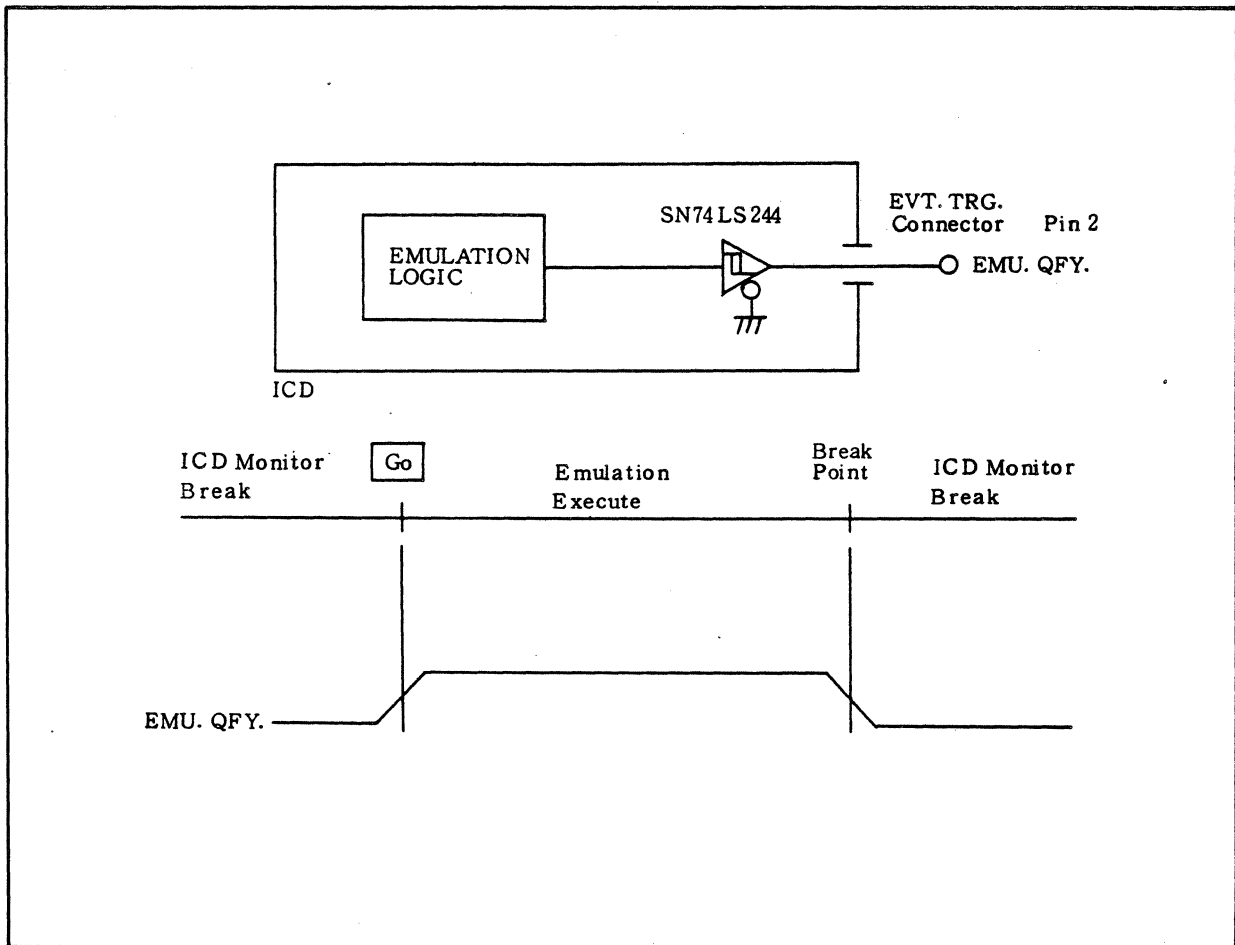


Figure 4-F.2. Emulation Qualify Output Timing Diagram

MAP CONTROL PROBE

The Map Control probe enables the user memory to be used over the entire memory area (64K bytes) when a TTL signal is received by the MAP CTL. probe (in-circuit mode I1). This input is ignored during an emulation break (Figure 4-F.3).

The Map Control probe must be set to a LOW level (minimum 125ns) before the RD or WR signal is generated (Figure 4-F.3).

The Map Control signal is useful to develop/debug a system which uses phantom ROM (ROM that operates for the system bootstrap procedure and then hides behind the main memory). If a program is executed when mapping the address of the phantom ROM into the ICD program memory, the ICD cannot select the user memory with same address as the phantom ROM. However, it is possible to input the signal of the main memory to the ICD's Map Control. The phantom ROM then uses the ICD program memory and the main memory accesses the user memory with the same address.

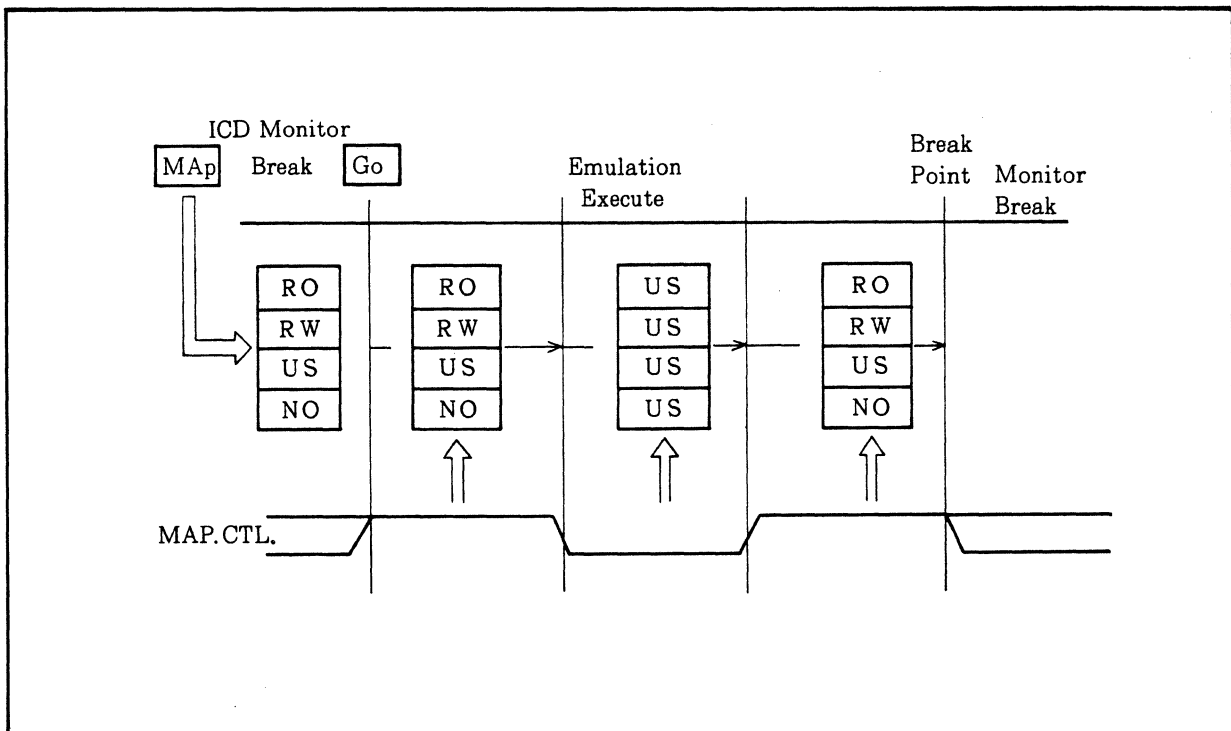


Figure 4-F.3. Map Control Timing Configuration

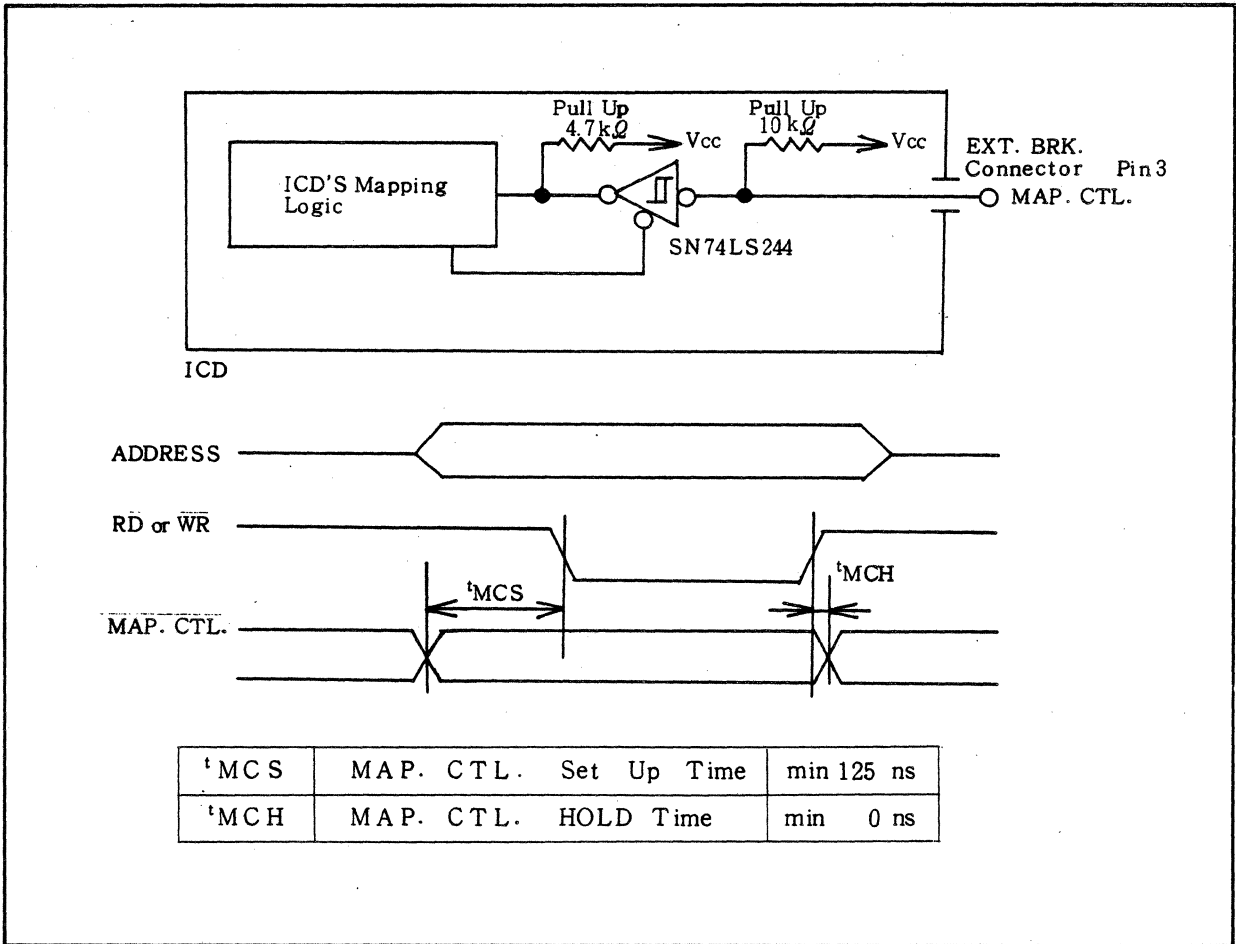


Figure 4-F.4. Map Control Timing Diagram

G CONTROL MODULES

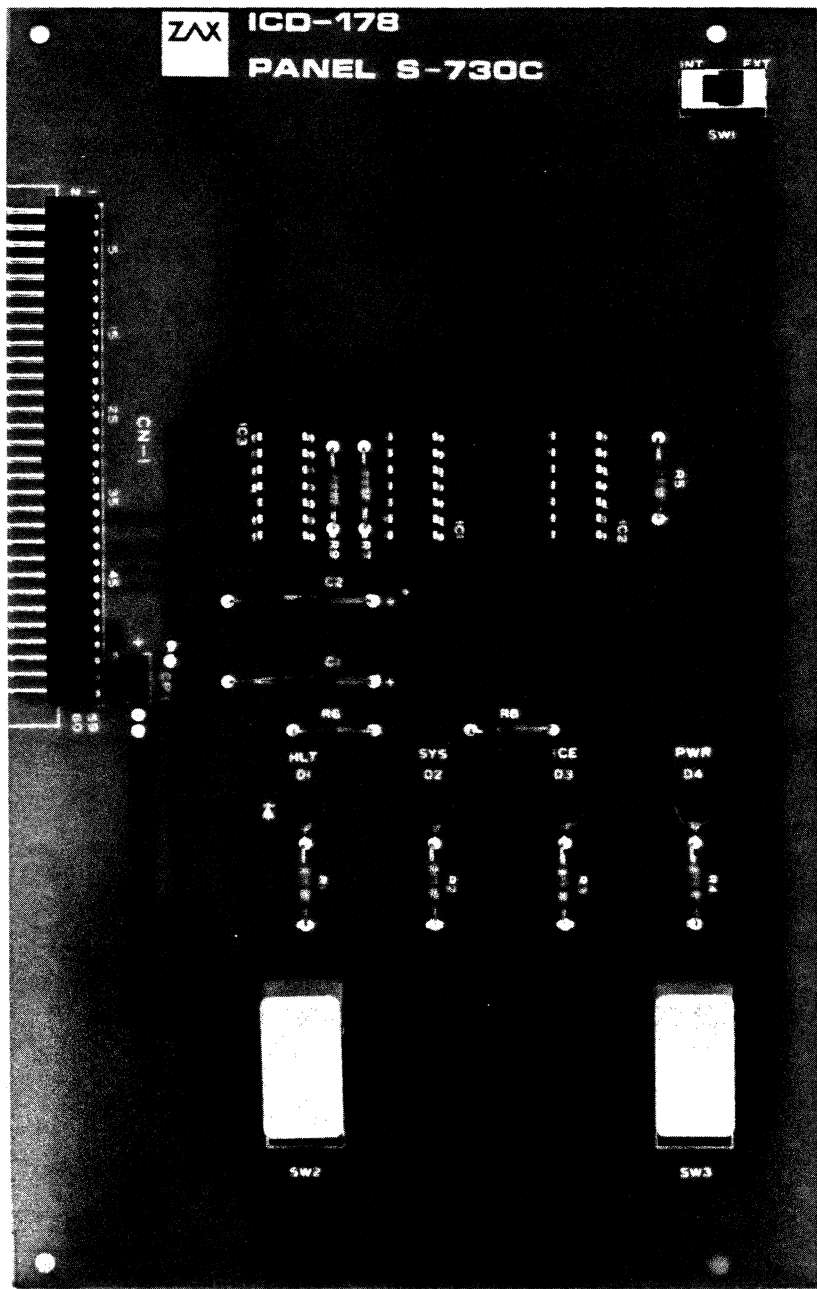
There are five control modules located within the ICD-278 in addition to the power supply. These modules are listed in their order of position starting from the top (Operator Panel) of the ICD;

PANEL	S-730C	Indicator Panel
*SIO	S-791B	Serial Interface (C)
RTS	S-795A	Realtime Trace
*CPU	S-793A	Central Processing Unit (E)
EMU	S-792A	Emulation Memory Unit

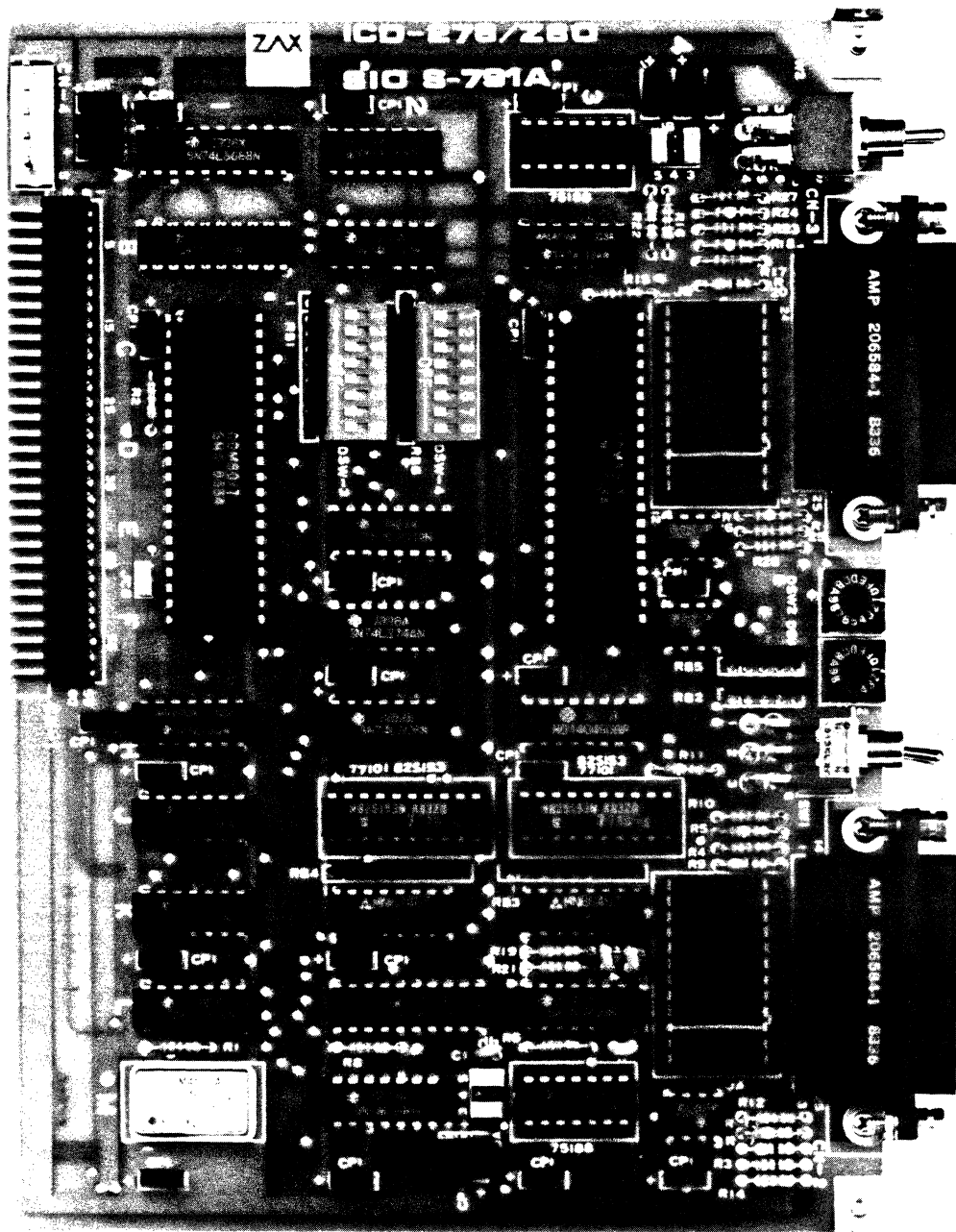
Note: These modules (*) are identified and described in detail elsewhere in this section. See the reference letter at the end of these module names for their location in this section or refer to the Table of Contents.

Each module is linked by the Mother Bus cable. Power is supplied to each module by a 5-pin, plug-type power connector cable (except for the S-730C module which uses the 60-pin Mother Bus connector cable).

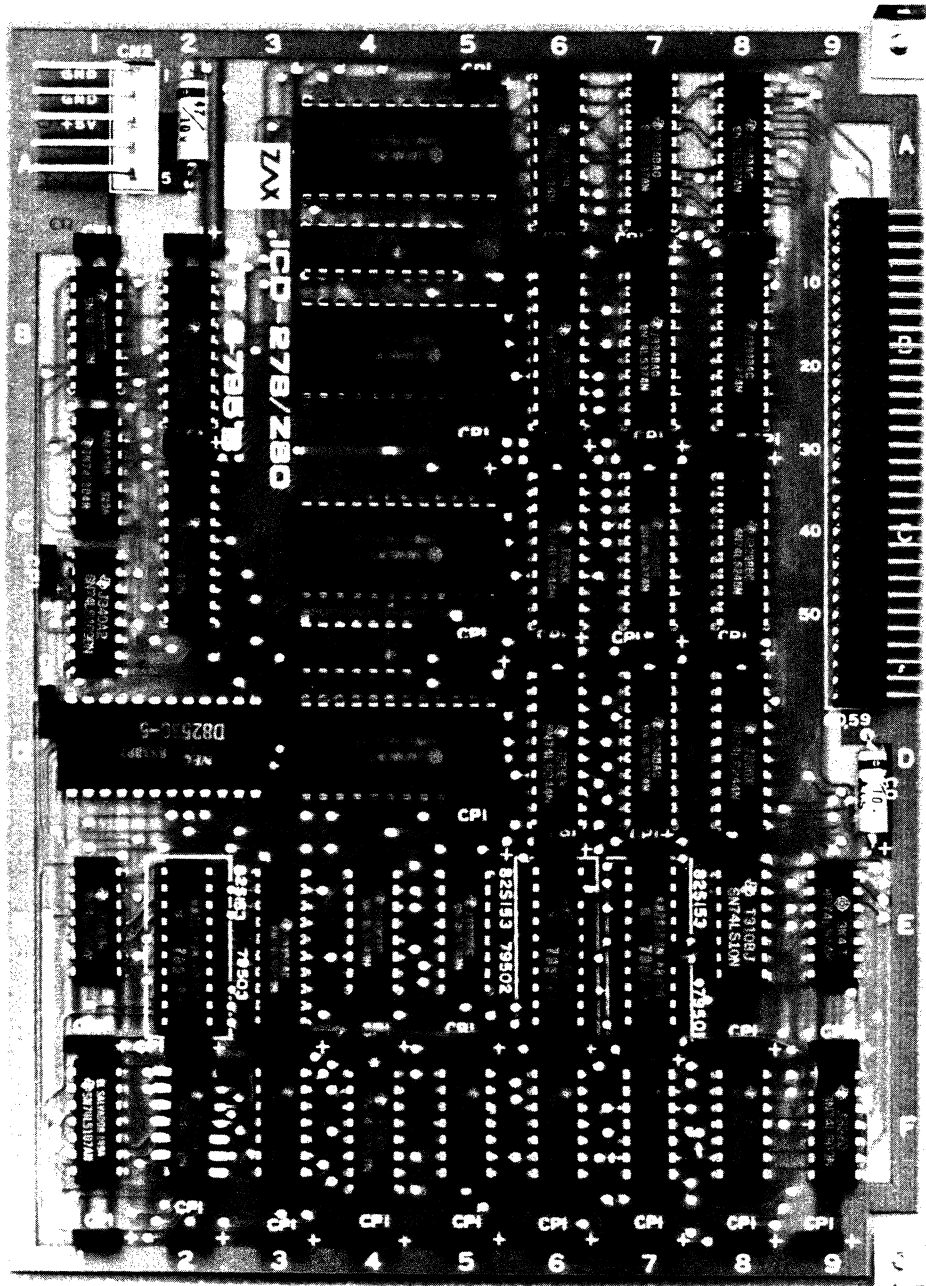
The modules are attached to the ICD chassis by brackets which are screw-mounted (two or four screws) to secure them in place. Removing any one of the control modules necessitates some disassembly of the ICD (see Disassembly Procedure located at the end of this chapter).



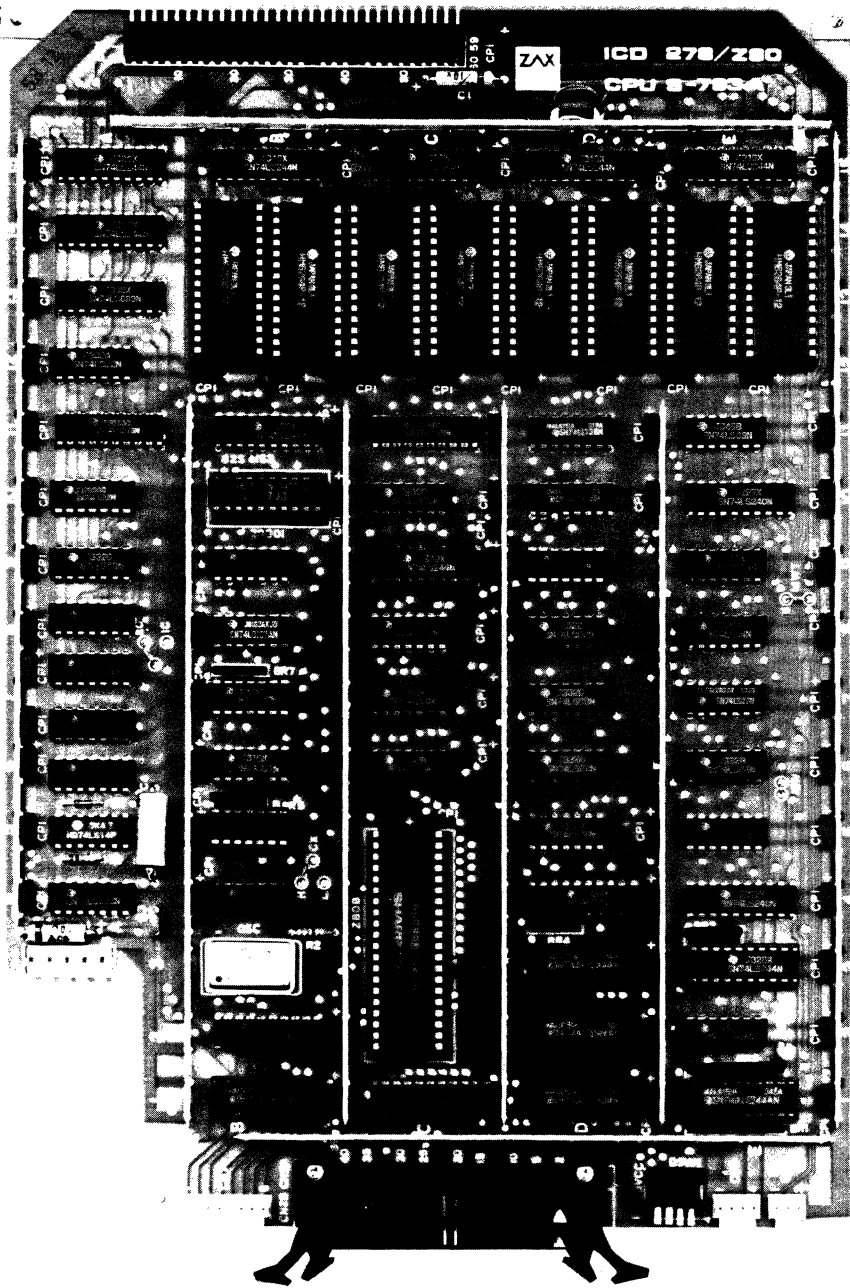
S-730 Indicator Panel



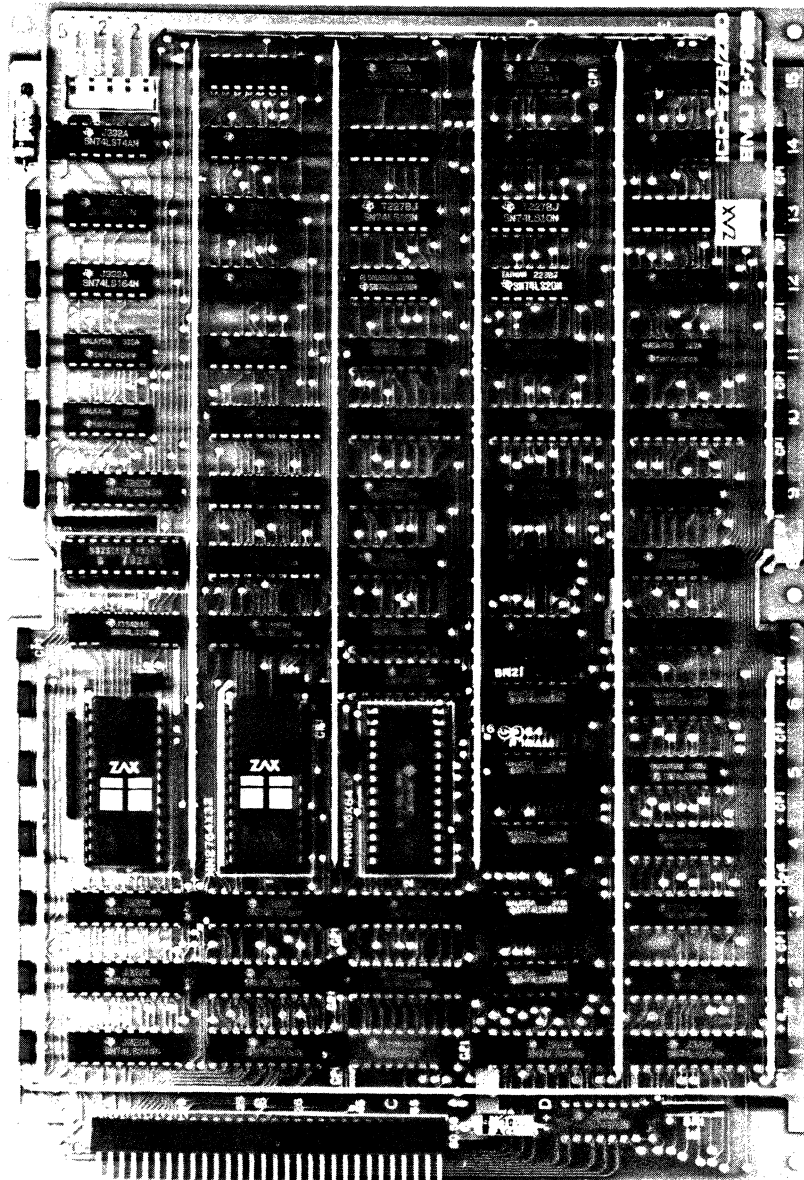
S-791 Serial Interface Output (SIO)



S-795 Realtime Trace Storage (RTS)



S-793 Central Processing Unit (CPU)



S-792 Emulation Memory Unit (EMU)

ICD DISASSEMBLY / MODULE REMOVAL

**WARNING: HAZARDOUS VOLTAGE IS PRESENT WITHIN THE ICD-278.
DISCONNECT THE AC POWER PLUG BEFORE BEGINNING ANY
INTERNAL WORK ON THE ICD-278.**

Disassembly Procedure

- 1) Begin disassembly by inverting the ICD-278, placing the Operator Panel side; down. Access to the bottom panel is now possible.

NOTE: Place the ICD-278 on a soft foam-type pad to protect the case and components.

Remove the four large screws that hold the bottom panel to the chassis but do not detach the panel. Remove the four counter-sunk screws at each end (eight total) that hold the side panels in place.

- 2) Gently turn over the ICD-278, placing the Operator Panel side; up. Remove the two round-head screws at each end (four total) that join the top and side panels. Detach the side panels and the top panel.

The control modules are now accessible for removal.

- 3) **IMPORTANT:** Note the position of the power connectors before removing. Both the socket and plug have a black-colored identification label for proper positioning during reconnection.

Free the power cable by disconnecting its five-pin socket (CN4) from the module then detach the mother bus cable from its location on the module (CN1).

- 4) Remove the screws which hold the module to the chassis:

A) The top and bottom (S-730C and S-795A) modules are mounted directly to the chassis with four and six screws respectively. Remove these screws to detach the modules.

B) The three remaining modules (S-791A, S-793A and S-792A) are connected to brackets which screw mount to the ICD chassis at one end and slide into holders at the other end. To detach these modules, remove the screws and then carefully slide the modules away from the chassis.

Module Installation

Installation is the reverse of removal.

CAUTION: DO NOT REVERSE POWER CONNECTOR POSITION DURING INSTALLATION. CONNECTOR MISPLACEMENT WILL CAUSE DAMAGE TO THE ICD-278.

NOTE: When replacing the side panels, position all the screws in place loosely to allow the panels to align properly before tightening.

PART II

SOFTWARE SPECIFICATION GUIDE

INTRODUCTION

Part II of this Operation Manual contains software information for the ICD-278. This information relates to the various software commands (including the Z commands) as well as the necessary protocol for communication with a host computer system.

A detailed analysis of the user commands is explained in Section V; Software Commands. Emulator handshaking with the host computer is described in Section VI; Host Computer System Communication.

Before beginning communication with the host computer, select and construct the proper system configuration (REMOTE or LOCAL: standalone with host computer). The two system configurations are detailed in Section III; System Configuration and Start-up Sequence, and reprinted in Section VI for convenience.

PART II SOFTWARE SPECIFICATION GUIDE	Page
--------------------------------------	------

Section V Software Commands	
-----------------------------	--

Load	5-5
Save	5-6
Verify	5-7
Quit	5-8
Z Log	5-8
Z Batch	5-8
Z Comment	5-9
Z Wait	5-9
Z Directory	5-10
Z Help	5-10
Z Repeat	5-11
Z Print	5-11

Section VI Host Computer System Communication	
---	--

Idle Program	6-4
Display Text	6-6
Command Input Request	6-8
Object File Load/Verify	6-10
Object File Save	6-12
Z Command	6-14
Quit	6-17
Display Symbolic Text	6-18
Console Key Check	6-20

THIS PAGE INTENTIONALLY BLANK

SECTION V

SOFTWARE COMMANDS

INTRODUCTION

This section explains the use of the ZICE utility software commands. The ZICE commands are used to interface the ICD-278 to other microprocessors and microcomputers for testing programs under development. Additionally, the ZICE utility supports four user commands; Load, Save, Verify and Quit. These four commands, including address parameters and application notes, are described in detail in Part III: COMMAND REFERENCE GUIDE.

ZICE KEYWORDS

The following user-supplied keywords are used with the ZICE commands;

portspec :specifies input/output port to consider, either;
 T (Terminal port/ignore software handshake)
 P (Terminal port/perform handshake)
 A (Auxiliary port)
 H (Host port)

filename :name of the operation file.

bias :hexadecimal (16-bit, 0-F) address that is added to start address of each program or data address.

message :input data, ASCII 32 or hexadecimal.

beg.addr :beginning address, physical 16-bit.

end addr :ending address, physical 16-bit or L byte.

user addr :starting address of user program, physical 16-bit or hexadecimal 16-bit address.

obj.name :file name, all standard CPM files are available (hex, assembly, etc.)

display :used with the Directory command, specifies the type of directory display.

range :an integer number used with the Repeat command.

name :the name of a specific command; Go, Map, etc.

nth-name :the last command name to be used during the repeat process.

comment :a displayed user supplied comment or statement.

ZICE FORMAT

Note the following format when entering the ZICE commands.

*Keywords/key characters are displayed as; **C / 9** etc, or as BOLD characters. They represent words/letters/characters which must be entered. Any combination of upper/lower case letters may be used.

*Lowercase italicized letters show items which the user must supply, such as "filename" in which the user would enter the name of a selected file.

*A vertical line "|" is shown to indicate a choice between separated information which must be selected, such as;

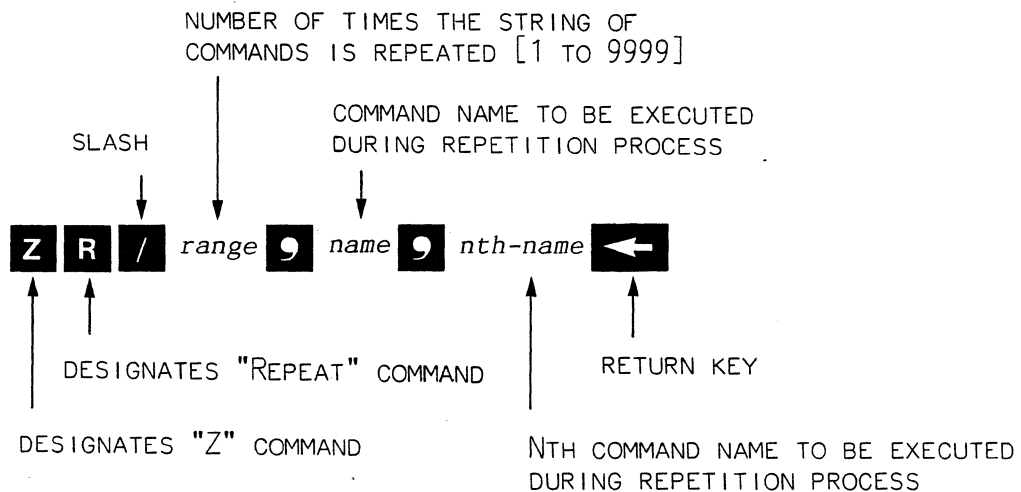
ON|OFF

ON or OFF may be entered, but not both.

*Include all punctuation such as commas, equal signs, colons, slashes, feature keys. Etc.

*Available address/data parameters are given at the end of the explanations and grouped as; [phy 16, L byte, ASCII 32, etc].

Example: REPEAT (R) command;



THIS PAGE INTENTIONALLY BLANK

THE LOAD COMMAND IS USED TO LOAD A HEXADECIMAL FILE ONTO A DISKETTE OR THE OBJECT PROGRAM (INTEL FORMAT), FROM EITHER SPECIFIED PORT IN MEMORY.

L / *portspec filename* **,** *bias* **,** *message* **←**

portspec - PORT SPECIFICATION;

T	:TERMINAL PORT (IGNORE SOFTWARE HANDSHAKE).
P	:TERMINAL PORT (PERFORM SOFTWARE HANDSHAKE).
A	:AUXILIARY PORT (IGNORE SOFTWARE HANDSHAKE).
H	:HOST PORT (PERFORM SOFTWARE HANDSHAKE).

filename :NAME OF THE FILE TO LOADED.

bias :BIAS OF THE OBJECT TO BE LOADED [L BYTE]
(DEFAULT = OBJECT STARTING ADDRESS).

message :LOAD MESSAGE [ASCII 32,HEX DATA].

EXAMPLE - LOAD FILE PROG3.HEX FROM DRIVE B
OF THE BOX;

L B:PROG3.HEX (CR)

THE SAVE COMMAND IS USED TO CREATE A USER PROGRAM ON A DISKETTE AS THE HEX FILE -OR- DUMP THE USER PROGRAM TO A SPECIFIED PORT USING THE INTEL FORMAT.

S **A** / *portspec filename* **,** *beg.addr* **,** *end addr* **,** *user addr* **,** *message* **←**

portspec - PORT SPECIFICATION;

T	: TERMINAL PORT (IGNORE SOFTWARE HANDSHAKE)
P	: TERMINAL PORT (PERFORM SOFTWARE HANDSHAKE)
A	: AUXILIARY PORT (IGNORE SOFTWARE HANDSHAKE)
H	: HOST PORT (PERFORM SOFTWARE HANDSHAKE)

filename : NAME OF FILE TO CREATE.

beg.addr : BEGINNING MEMORY ADDRESS [PHY 16].

end addr : ENDING MEMORY ADDRESS [PHY 16].

user addr : BEGINNING USER PROGRAM ADDRESS [PHY 16, HEX].

message : SAVE MESSAGE [ASCII 32, HEX DATA].

EXAMPLE - GENERATE THE TEST.HEX FILE ON DISKETTE;

>SA TEST,0,37FF,0

THE VERIFY COMMAND IS USED TO COMPARE A HOST COMPUTER FILE OR OBJECT PROGRAM (INTEL FORMAT) WITH THE CONTENTS IN MEMORY.

V / *portspec* **,** *obj.name* **,** *bias* **,** *message* **←**

portspec - PORT SPECIFICATION;

T :TERMINAL PORT (IGNORE SOFTWARE HANDSHAKE)
P :TERMINAL PORT (PERFORM SOFTWARE HANDSHAKE)
A :AUXILIARY PORT (IGNORE SOFTWARE HANDSHAKE)
H :HOST PORT (PERFORM SOFTWARE HANDSHAKE)

obj.name :OBJECT NAME TO COMPARE.

bias :BIAS OF THE FILE OR OBJECT TO COMPARE
(DEFAULT = OBJECT STARTING ADDRESS).

message :VERIFY MESSAGE [ASCII 32, HEX DATA].

EXAMPLE - COMPARE XASM WITH MEMORY;


V XASM,(CR)

.ADRS	M	0
020F	0B	67
105C	78	00
105D	B7	00
0100		

THE LOG SUBCOMMAND ECHOES THE CONSOLE OUTPUT TO A DISK FILE (ON) OR TURNS OFF THE ECHO FEATURE (OFF).

Z **L** ON|OFF 

THE BATCH SUBCOMMAND IS USED TO LOAD AND EXECUTE A FILE CONTAINING ZICE COMMANDS WHICH WERE PREVIOUSLY CREATED USING THE EDITOR. THE COMMANDS ARE EXECUTED IN THE ORDER IN WHICH THEY APPEAR IN THE TEXT FILE. A CTRL-C INTERRUPTS THE COMMAND SEQUENCE DURING THE EXECUTION OF A BATCH COMMAND FILE.


Z **B** *filename* 

filename :SPECIFIES THE FILE NAME

THE QUIT COMMAND IS USED TO TERMINATE THE ICD-278 EMULATOR AND RETURN CONTROL BACK TO THE HOST COMPUTER SYSTEM.

Q 

THE COMMENT SUBCOMMAND IS USED TO OUTPUT COMMENTS DURING A SUBCOMMAND PROCESS.

Z **C** **,** *comment* 

comment :COMMENT STRING WHICH APPEARS ON THE CONSOLE DURING THE EXECUTION OF THE BATCH OR REPEAT PROCESS.

EXAMPLE;

```
Z C, *** FD TEST START ***  
L FDTEST  
G O, FF  
Z C, *** MEMORY TEST START ***  
L MENTEST
```

THE WAIT SUBCOMMAND IS USED TO PLACE A TEMPORARY PAUSE IN A REPEAT OR BATCH SUBCOMMAND PROCESS. THE EXECUTION IS THEN HALTED UNTIL THE 'SPACE' BAR IS PRESSED - RESTARTING THE PROCESS.

Z **W** **,** *comment* 


comment :FIELD TO BE USED FOR DOCUMENTATION OF A WAIT COMMAND.

EXAMPLE;

```
Z R, 10/G X,X/D X,X/Z W(CR)
```

```
.REPEAT PROCESS TEN TIMES  
EXECUTE INSTRUCTION  
DISPLAY RESULTS  
WAIT FOR OPERATOR RESPONSE
```

THE DIRECTORY SUBCOMMAND IS USED TO DISPLAY A DISKETTE DIRECTORY WHILE USING THE ZICE UTILITY.

Z D *display* 

display - SPECIFIES;


:DISPLAY THE DIRECTORY OF HEXADECIMAL FILES.

:DISPLAY THE LIST OF ALL .BAT FILES.

:DISPLAY A LIST OF ALL .HEX FILES
(DEFAULT PARAMETER).

:DRIVE ON WHICH THE DIRECTORY RESIDES.

THE HELP SUBCOMMAND IS USED TO DISPLAY THE FORMAT OF A COMMAND BY SPECIFYING A PARTICULAR COMMAND -OR- DISPLAYS THE COMPLETE COMMAND LISTING

Z H *name* 

name :COMMAND FOR WHICH THE HELP REQUEST IS
DIRECTED.

EXAMPLE;

Z H,V (CR)

V I/O, FILE NAME (CR)

THE REPEAT SUBCOMMAND IS USED TO PROVIDE A METHOD OF REPEATING A COMMAND OR A STRING OF COMMANDS. THIS SUBCOMMAND IS USEFUL WHEN TESTING SUBROUTINES AS SHOWN IN THE EXAMPLE.

Z R 9 *range* / *name* / *nth-name* ←

range :THE NUMBER OF TIMES THE STRING OF COMMANDS ARE TO BE REPEATED. THE RANGE OF N IS 1 TO 9999 (DEFAULT IS INFINITE REPETITION).

name :COMMAND NAME TO BE EXECUTED DURING THE REPETITION PROCESS.

nth-name :NTH COMMAND NAME TO BE EXECUTED DURING THE REPETITION PROCESS.

EXAMPLE;

Z R, 427/G X,X/Z P,ON/D X,X/Z P, OFF/G X,X/ (CR)

REPEAT 427 TIMES
EXECUTE INSTRUCTIONS
PRINT CONSOLE MESSAGES
DISPLAY SUBROUTINE RESULTS
TURN OFF PRINTING CONSOLE MESSAGES
EXECUTE INSTRUCTION

THE PRINT SUBCOMMAND PROVIDES THE USER WITH A HARDCOPY OF THE COMMAND DIALOG AND DATA DISPLAYED ON THE CONSOLE DURING A ZICE UTILITY SESSION (ON) OR TURNS OFF THE HARDCOPY PRINT FEATURE (OFF) [DEFAULT CONDITION AT INITIALIZATION.

Z P ON|OFF ←

THIS PAGE INTENTIONALLY BLANK

SECTION VI

HOST COMPUTER SYSTEM COMMUNICATION

INTRODUCTION

This section details the communication protocol to be observed when interacting the ICD-278 with a host computer system. Seven communication sequences are illustrated by protocol diagrams showing program analysis and flowcharts. Additionally, Idle program and Function Analyze program flowcharts are included.

Both Local and Remote mode configurations are examined.

LOCAL Mode

The Local (standalone with host computer) mode allows object I/O between the ICD-278 and the host computer system. Once configured, the ICD-278 operation commands are executed through the console terminal. The Local mode enables the user commands; Load, Save, Verify and Quit. Object downloading from the host computer to the ICD-278 maybe accomplished in this mode also.

REMOTE Mode

The Remote mode allows file transfer between the ICD-278 and host computer via the support software program of the host computer system. It is also possible to add the batch processing function for batch files and to use symbolic debugging in this mode.

Before data transfer begins, proper ICD system configuration must be selected and constructed. The LOCAL and REMOTE (standalone with host computer) modes are illustrated here (Figure 6-1). A detailed explanation of system configuration and start-up procedure may be found in Section III.

COMMUNICATION TERMINOLOGY

The communication sequences used in this section are illustrated by protocol diagrams with definitions. The following data communication terminology is used in the diagram explanations;

- SOH - Start Of Heading
Precedes a block of heading characters which provide auxiliary information, such as routing and priority.
- ACK - Acknowledge
An acknowledge to indicate the successful reception of a message segment.
- NAK - Not Acknowledge
An acknowledge to indicate the unsuccessful reception of a message segment.
- STX - Start of Text
- ETX - End of Text
- TEXT - That part of the message which contains the substantive information to be conveyed.
- CR - Call Request or Carriage Return

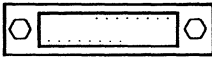
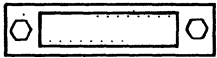
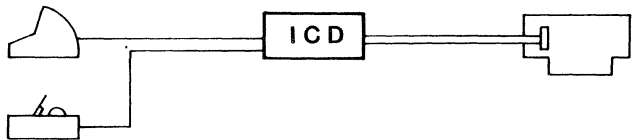

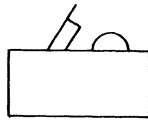
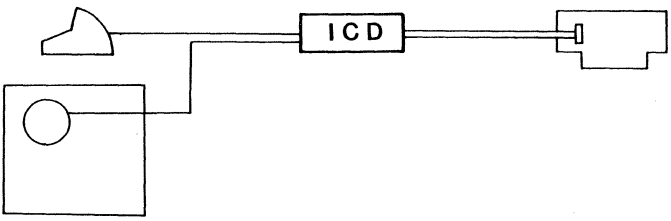
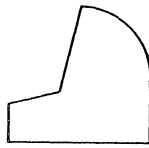
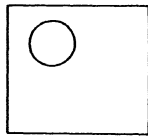
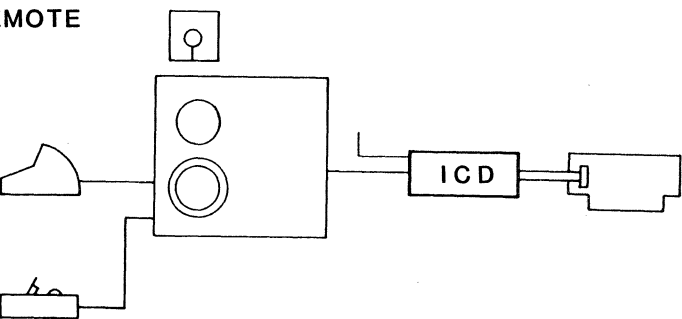
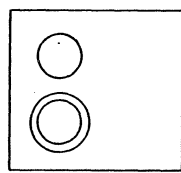
SYSTEM CONFIGURATION	▼ LOCAL <input type="checkbox"/> REM ▲	DCE <input type="checkbox"/> DTE	TERMINAL 	HOST/AUX 
LOCAL (standalone) 	▼ LOCAL <input checked="" type="checkbox"/> REM ▲	DCE <input checked="" type="checkbox"/> DTE	 CONSOLE	 PRINTER
LOCAL (standalone with Host Computer) 	▼ LOCAL <input checked="" type="checkbox"/> REM ▲	DCE <input checked="" type="checkbox"/> DTE	 CONSOLE	 HOST SYSTEM
REMOTE 	▼ LOCAL <input checked="" type="checkbox"/> REM ▲	DCE <input checked="" type="checkbox"/> DTE		 HOST SYSTEM

Figure 6-1. System Configuration

IDLE PROGRAM

IDLE PROGRAM

- 1) THE HOST SYSTEM WAITS FOR AN INPUT FROM THE ICD. THE HOST SYSTEM IS PROVIDED WITH AN INPUT BUFFER FOR INPUT CODES FROM THE ICD.
- 2) THE HOST SYSTEM ENTERS THE CODES RECEIVED FROM THE ICD IN THE INPUT BUFFER.

IF RECEIVED CODE IS;

<CR>

<SOH>

HOST SYSTEM EXECUTES;

DISPLAY TEXT PROGRAM

FUNCTION ANALYZE PROGRAM

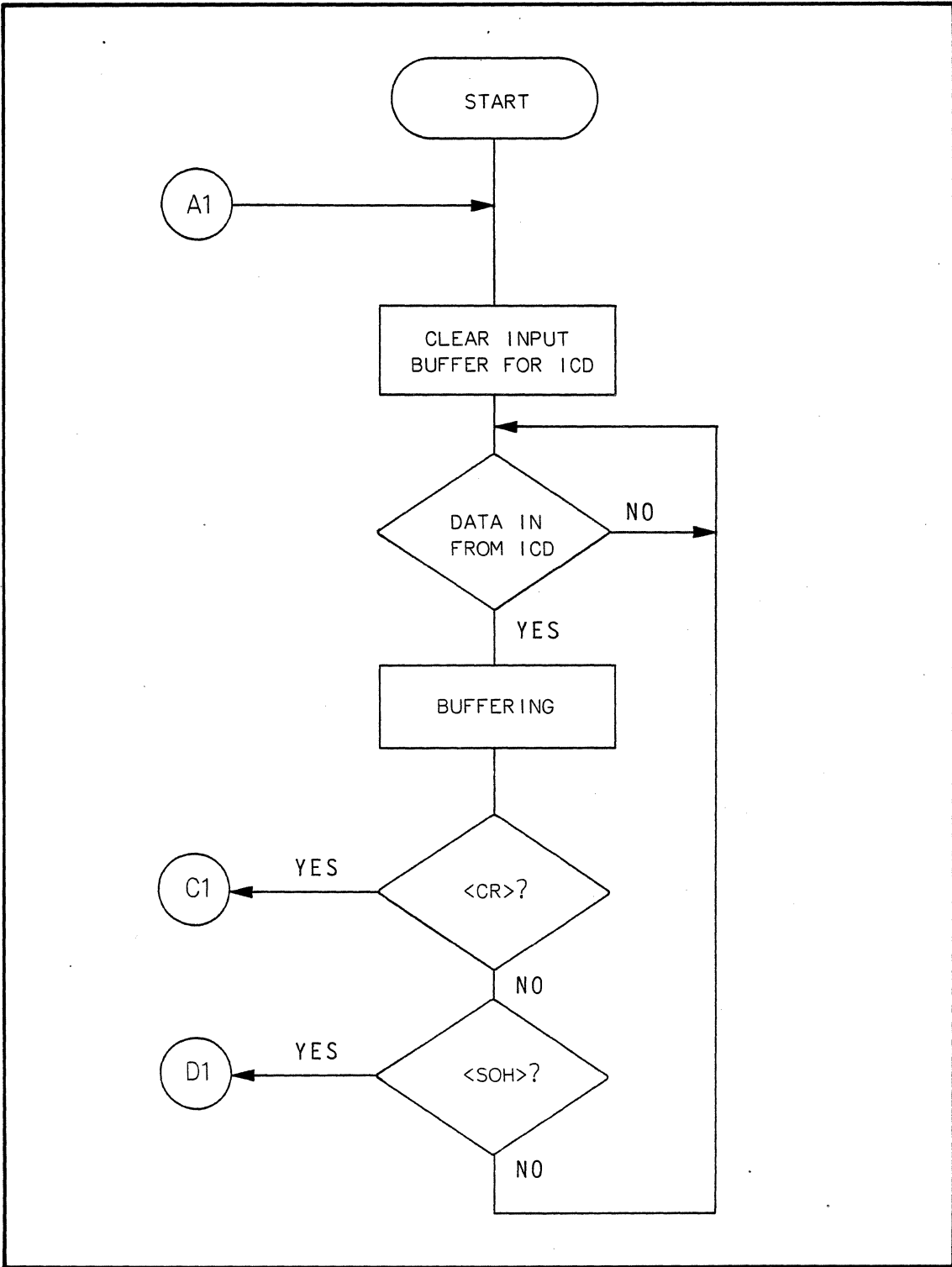


Figure 6-2. Idle Flowchart

DISPLAY TEXT

THE DISPLAY TEXT SEQUENCE IS USED TO DISPLAY TEXTS, SENT FROM THE ICD, ON THE HOST COMPUTER CONSOLE.

- (TEXT) <CR> : TEXT RECORD
THIS RECORD IS THE ASCII TEXT SENT TO THE HOST SYSTEM TO BE DISPLAYED ON THE HOST CONSOLE. TEXT RECORDS MAY NOT INCLUDE; <ACK>, <NAK> OR <SOH> CONTROL CODES. THE HOST SYSTEM DISPLAYS A LINE UP TO <CR> WHENEVER THE TEXT RECORD ON THE HOST SYSTEM IS DISPLAYED. AFTER DISPLAYING A LINE OF RECORDS, THE CURSOR MOVES TO THE NEXT LINE.
- <ACK> : DISPLAY END CODE
THIS CODE IS SENT TO THE ICD WHEN NO MORE DISPLAY TEXT RECORD EXISTS.
- <NAK> : DISPLAY HALT CODE
THIS CODE IS SENT TO THE ICD TO HALT THE TEXT RECORD TRANSMISSION DURING THE DISPLAY OF 'DUMP' 'TRACE' ETC.

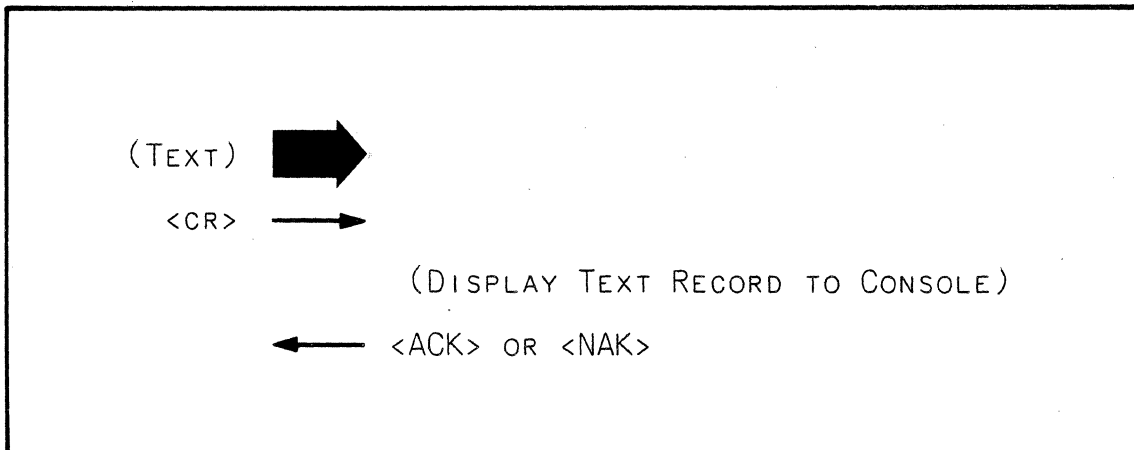


Figure 6-3. Display Text Sequence

DISPLAY TEXT PROGRAM

- 1) THE ICD SENDS LF(TEXT)<CR>LF TO THE HOST SYSTEM WHEN A TEXT IS DISPLAYED.
- 2) IF A TEXT 1 RECORD IS RECEIVED FROM THE ICD, THE HOST SYSTEM DISPLAYS THE TEXT 1 RECORD ON THE HOST SYSTEM CONSOLE.
- 3) THE HOST SYSTEM NOW CHECKS THE INPUT STATE OF THE HOST SYSTEM CONSOLE;
 - A) IF THERE IS NO INPUT FROM THE HOST SYSTEM CONSOLE, THE HOST SYSTEM WILL SEND <ACK> TO THE ICD AND THEN RETURN TO THE IDLE PROGRAM.
 - B) IF THERE IS INPUT FROM THE HOST SYSTEM CONSOLE AND THE INPUT CODE IS <ESC>, THE HOST SYSTEM WILL SEND <NAK> TO THE ICD AND THEN RETURN TO THE IDLE PROGRAM.
 - C) IF THERE IS INPUT FROM THE HOST SYSTEM CONSOLE BUT THE INPUT CODE IS NOT <ESC>, THE HOST SYSTEM WILL SEND <ACK> TO THE ICD AND THEN RETURN TO THE IDLE PROGRAM. IN THIS MODE, ONLY THE <ESC> KEY (DISPLAY HALT) IS EFFECTIVE.

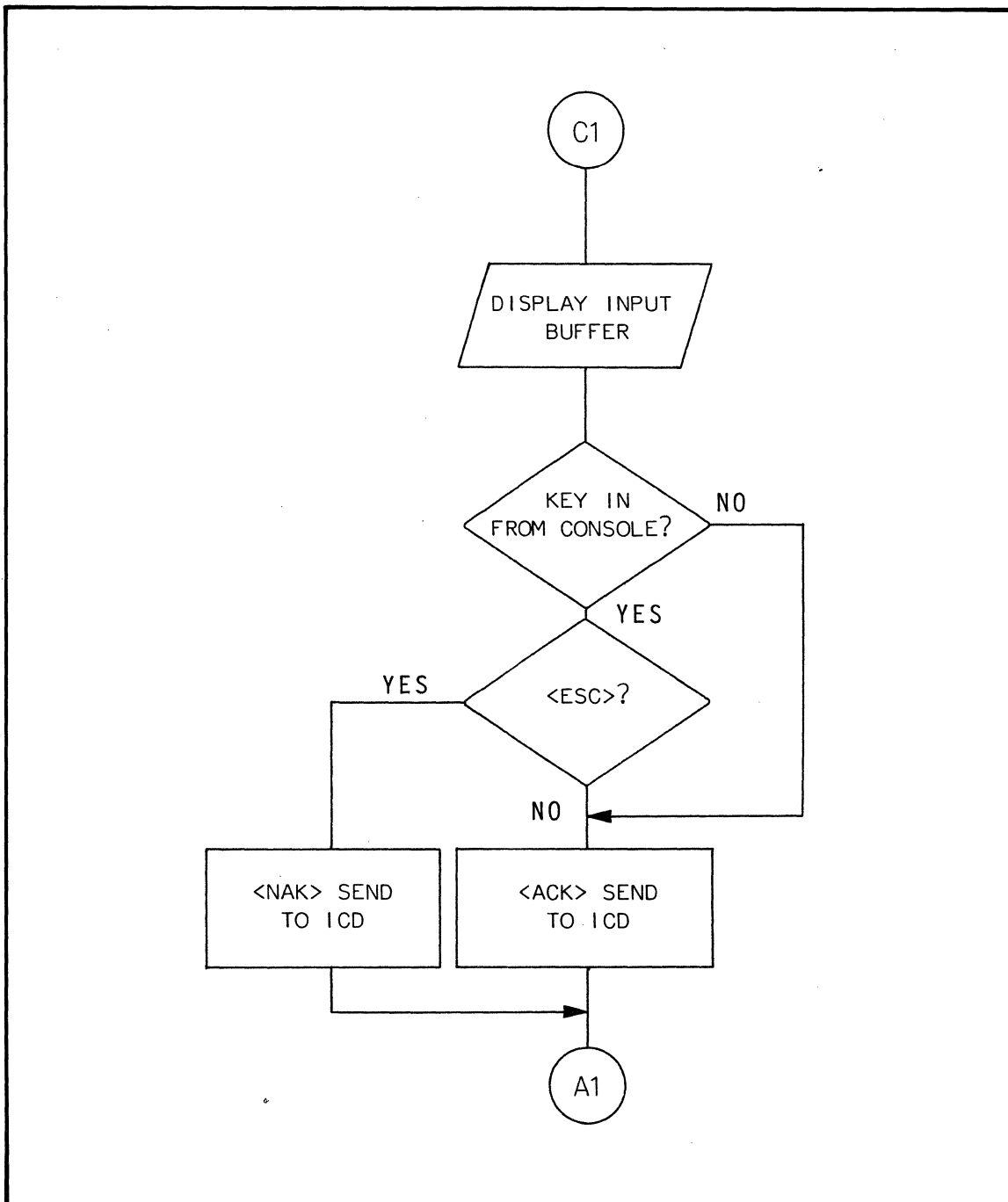


Figure 6-4. Display Text Flowchart

COMMAND INPUT REQUEST PROGRAM

COMMAND INPUT REQUEST SEQUENCE

(TEXT)<ACK> : COMMAND REQUEST RECORD
THIS RECORD IS SENT WHEN THE ICD HAS REQUESTED A COMMAND FROM THE HOST SYSTEM. THE COMMAND REQUEST RECORD ALSO INCLUDES AN ASCII TEXT TO DISPLAY ON THE HOST SYSTEM CONSOLE. COMMAND REQUEST RECORDS MAY NOT INCLUDE; <ACK>, <NAK>, <CR> OR <SOH> CONTROL CODES.

IF THE HOST SYSTEM DISPLAYS A TEXT ON THE HOST SYSTEM CONSOLE, THE DATA IN THE INPUT BUFFER (SENT FROM ICD) WILL BE DISPLAYED UNTIL <ACK> IS ISSUED.

THE HOST SYSTEM MAY NOT OUTPUT DATA ON THE HOST SYSTEM CONSOLE. AFTER THE TEXT IS DISPLAYED, THE CURSOR MUST REMAIN AT THAT POSITION.

(COMMAND)<CR> : COMMAND RECORD
THIS RECORD IS A TEXT THAT IS SENT FROM THE HOST SYSTEM TO THE ICD AS A COMMAND. THE COMMAND RECORD MAY NOT BE TERMINATED WITH <CR> - NOR MAY IT INCLUDE ANY CONTROL CODES.

WHEN THE HOST SYSTEM ENTERS A COMMAND RECORD THROUGH THE HOST SYSTEM CONSOLE, A LINE OF DATA (UP TO <CR>) IS ENTERED WHICH IS DISPLAYED ON THE CONSOLE WITH ECHOBACK. AFTER A LINE OF COMMAND RECORD IS DISPLAYED, THE CURSOR IS MOVED TO THE NEXT LINE.

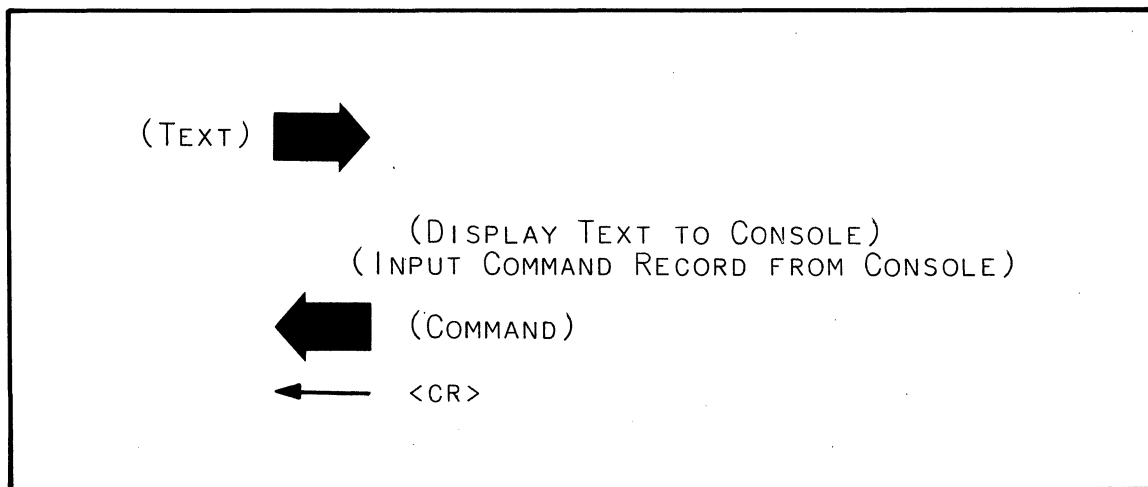


Figure 6-5. Command Input Request Sequence

COMMAND REQUEST PROGRAM

- 1) THE ICD SENDS LF(TEXT)<ACK>LF TO THE HOST SYSTEM IN RESPONSE TO A COMMAND REQUEST.
- 2) IF <ACK> IS RECEIVED FROM THE ICD, THE HOST SYSTEM WILL DISPLAY THE DATA IN THE INPUT BUFFER (SENT FROM THE ICD) ON THE HOST SYSTEM CONSOLE AND THEN WAIT FOR INPUT FROM THE HOST CONSOLE.
- 3) IF A COMMAND 1 RECORD IS INPUT FROM THE HOST SYSTEM CONSOLE, THE HOST SYSTEM WILL SEND THE 1 RECORD OF THE INPUT COMMAND TO THE ICD, AND THEN RETURN TO THE IDLE PROGRAM.

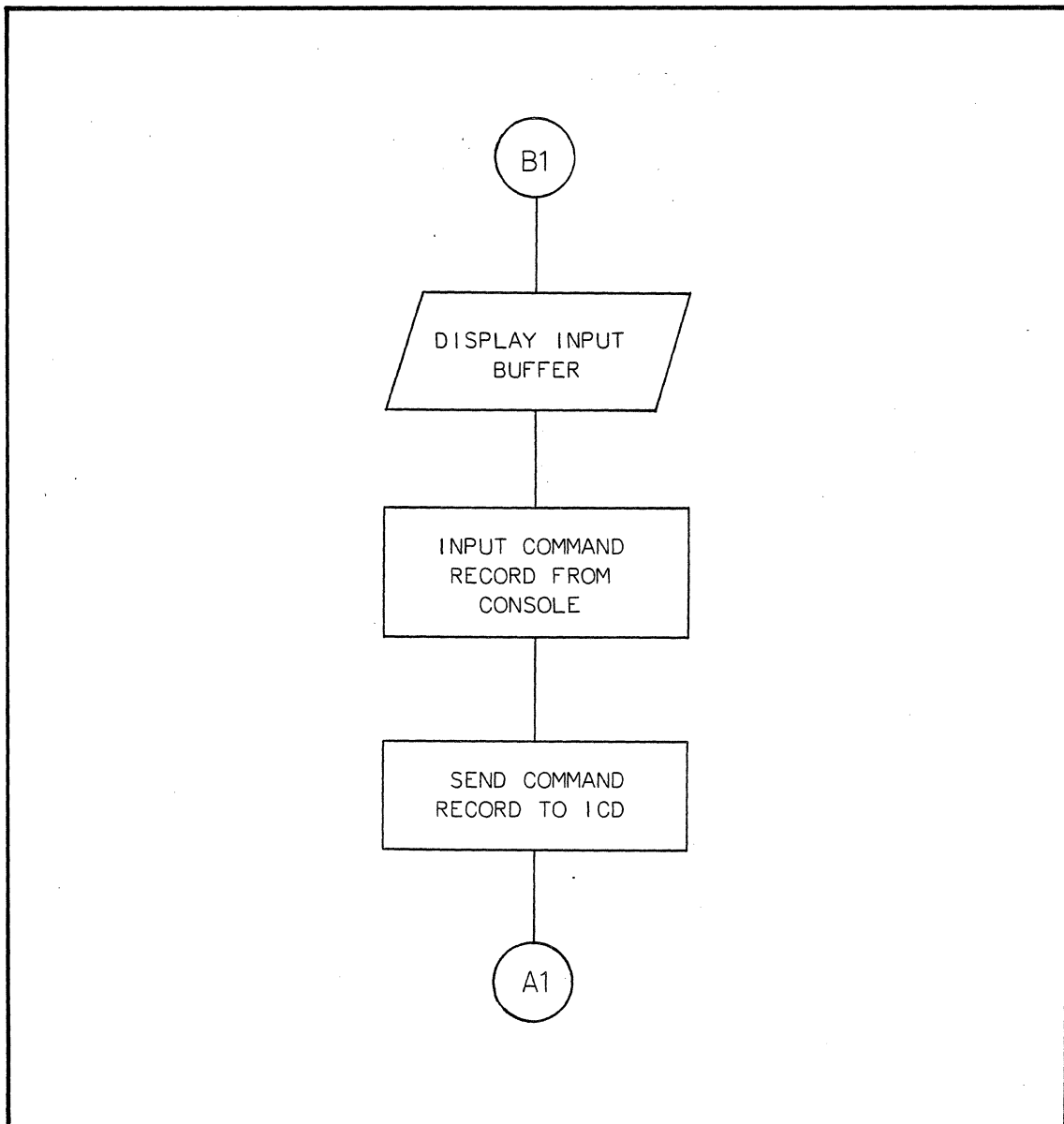


Figure 6-6. Command Request Flowchart

OBJECT FILE LOAD/VERIFY

OBJECT FILE LOAD/VERIFY SEQUENCE

THE FOLLOWING IS THE SEQUENCE FOR TRANSMISSION OF THE OBJECT FILE FROM THE HOST COMPUTER SYSTEM IN RESPONSE TO THE LOAD/VERIFY REQUEST FROM THE ICD.

NOTE: DURING OBJECT FILE TRANSMISSION, THE TEXT DISPLAY REQUEST FROM THE ICD OR CONSOLE KEY CHECK SEQUENCE, MAY BE INSERTED.

<SOH>"00"(FILE NAME)<CR> : LOAD REQUEST RECORD

THIS RECORD TRANSMITS TO THE HOST SYSTEM WHENEVER THE ICD LOADS THE OBJECT FILE. IT IS POSSIBLE TO USE THE (FILE NAME) FIELD AS THE USER DEFINITION LOAD MESSAGE.

<SOH>"01"(FILE NAME)<CR> : VERIFY REQUEST RECORD

THIS RECORD TRANSMITS TO THE HOST SYSTEM A VERIFICATION OF THE OBJECT FILE AND MEMORY SENT FROM THE ICD. IT IS POSSIBLE TO USE THE (FILE NAME) FIELD AS THE USER DEFINITION VERIFY MESSAGE.

(INTEL-HEX 1 RECORD)<CR> : OBJECT FILE RECORD

THIS INTEL-HEX 1 RECORD IS TRANSMITTED TO THE ICD FROM THE HOST SYSTEM. TERMINATION AT THE END OF THE OBJECT FILE RECORD MUST BE SPECIFIED WITH <CR>. ADDITIONALLY, THE CONTROL RECORD MUST NOT ENTER INTO OBJECT FILE CODE.

<ACK> : OBJECT FILE REQUEST CODE

THIS CODE IS TRANSMITTED WHEN THE ICD MAKES A REQUEST FOR THE INTEL-HEX 1 RECORD FROM THE HOST SYSTEM.

<NAK> : OBJECT FILE RETRANSMISSION REQUEST CODE

THIS CODE IS TRANSMITTED WHEN THE ICD REQUESTS THE OBJECT FILE TRANSMISSION IN REGARD TO THE HOST SYSTEM. RETRANSMISSION REQUESTS ARE PRIMARILY FOR MEMORY WRITE-IN ERRORS WHICH OCCUR DURING LOADING. AS A RESULT, THE HOST SYSTEM WILL NOT CARRY OUT THE RETRANSMISSION, BUT WILL TRANSMIT <NAK> TO THE ICD AND HALT THE OBJECT LOAD/VERIFY SEQUENCE.

(TEXT)<CR> : TEXT RECORD

THE TEXT RECORD IS PRIMARILY A VERIFY ERROR MESSAGE WHICH OCCURS DURING VERIFY. HOWEVER, IF THE ICD IS CONFIGURED IN THE LOCAL MODE, NO TEXT RECORDS ARE TRANSMITTED TO THE HOST SYSTEM (SEE TEXT DISPLAY SEQUENCE).

<ACK> : DISPLAY TERMINATION CODE

(SEE TEXT DISPLAY SEQUENCE)

<NAK> : LOAD/VERIFY SEQUENCE - HALT INSTRUCTION CODE

IF THE HOST SYSTEM TRANSMITS <NAK> TO THE ICD IN REGARD TO A TEXT RECORD TRANSMISSION, THIS CODE EFFECTS A HALT IN THE LOAD/VERIFY SEQUENCE.

<SOH>"09"<CR> : CONSOLE KEY INPUT CHECK REQUEST CODE

THE PRIMARY PURPOSE OF THIS CODE IS TO SUSPEND THE VERIFY ERROR MESSAGE DISPLAY -OR- AS THE RESUMPTION OF AN INQUIRY. IN THE ICD LOCAL MODE, THIS REQUEST IS NOT POSSIBLE (SEE CONSOLE KEY CHECK SEQUENCE).

<ACK> : CONSOLE CODE MINUS INPUT

(SEE CONSOLE KEY CHECK SEQUENCE)

<ACK> : CONSOLE INPUT CODE

ASCII CODE

(SEE CONSOLE KEY CHECK SEQUENCE)

<NAK> : LOAD/VERIFY SEQUENCE HALT INSTRUCTION CODE

CONSOLE KEY

INPUT CHECK

HALTS THE OBJECT LOAD/VERIFY SEQUENCE WHEN HOST SYSTEM TRANSMITS <NAK> TO ICD IN REGARD TO A CONSOLE KEY INPUT CHECK.

<ETX> : LOAD/VERIFY TERMINATOR CODE

THIS CODE IS TRANSMITTED TO THE ICD AFTER THE HOST SYSTEM CLOSES A FILE. IT IS USED WHEN A FILE RECORD TRANSMISSION TO THE ICD CEASES. OBJECT LOAD/VERIFY SEQUENCE IS TERMINATED.

<NAK> : LOAD/VERIFY SEQUENCE - HALT INSTRUCTION CODE

THIS CODE INSTRUCTS THE ICD TO HALT THE OBJECT LOAD/VERIFY SEQUENCE. THE HALT CODE IS USED WHEN THE HOST SYSTEM HAS DETECTED FILE READ ERRORS, ICD AND HANDSHAKE ERRORS, ETC. THE HALT INSTRUCTION CODE MAY BE TRANSMITTED AT ANY TIME.

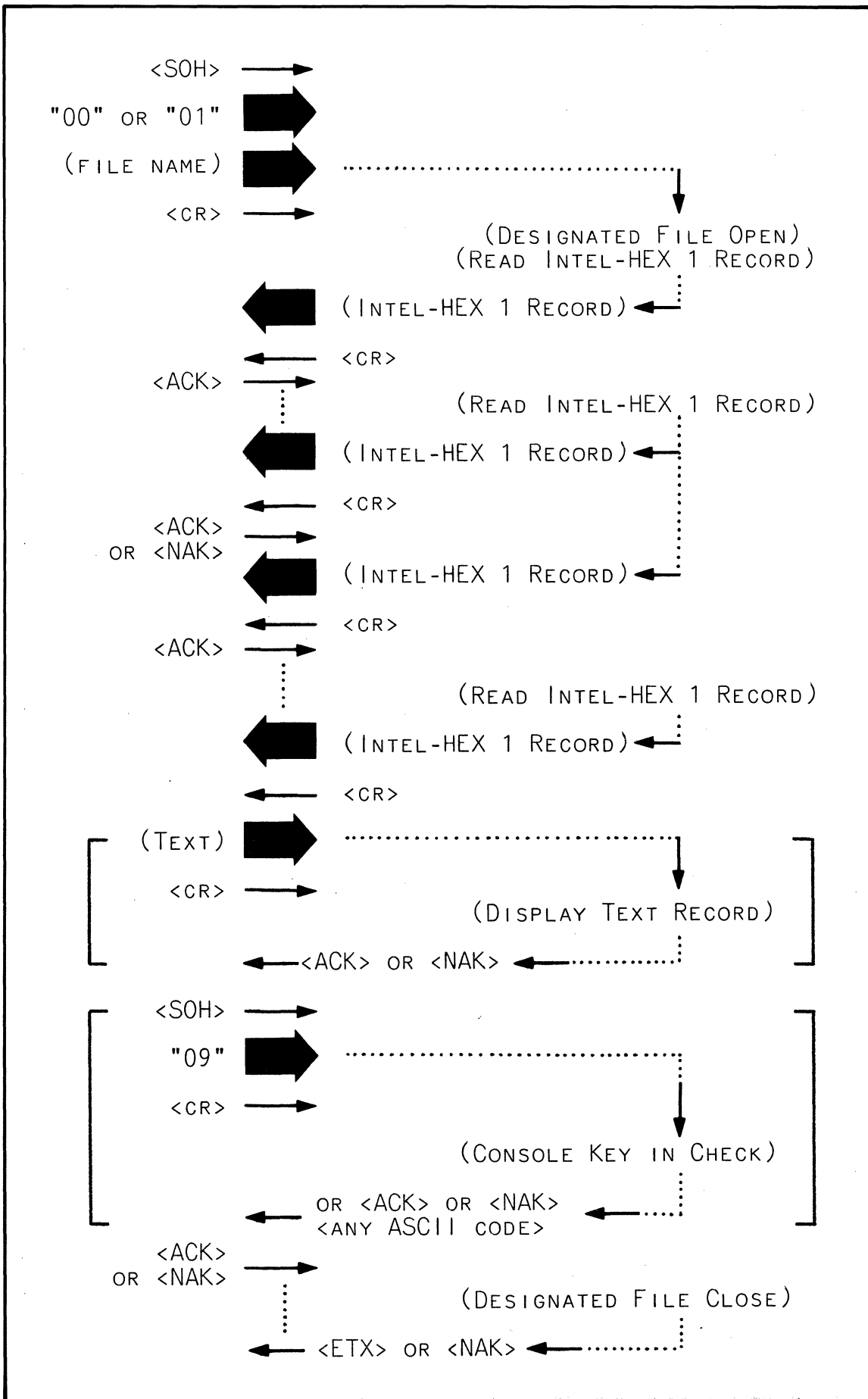


Figure 6-7. Object File Load/Verify Sequence

OBJECT FILE SAVE PROGRAM

OBJECT FILE SAVE SEQUENCE

THE FOLLOWING IS THE SEQUENCE FOR TRANSMISSION OF THE OBJECT FILE FROM THE HOST COMPUTER SYSTEM IN RESPONSE TO THE SAVE REQUEST FROM THE ICD.

<SOH>"02"(FILE NAME)<CR> : SAVE REQUEST CODE

THIS RECORD TRANSMITS TO THE HOST SYSTEM WHENEVER THE ICD SAVES A FILE. IT IS POSSIBLE TO USE THE (FILE NAME) FIELD AS THE USER DEFINITION SAVE MESSAGE.

<ACK> : OBJECT FILE REQUEST CODE
FILE WRITE

THIS CODE IS TRANSMITTED WHEN THE HOST SYSTEM REQUESTS A HALT IN THE OBJECT SAVE SEQUENCE. THE OBJECT SAVE REQUEST IS USEFUL FOR EITHER A TARGET MEMORY TIMING OUTPUT, OR NON-MEMORY ACCESS. AFTER THE HOST SYSTEM CLOSES THE FILE, <ACK> IS TRANSMITTED AND HALTS THE OBJECT SAVE SEQUENCE.

(INTEL-HEX 1 RECORD)<CR> : OBJECT FILE RECORD

THIS IS AN INTEL-HEX 1 RECORD RECEIVED BY THE ICD.

<ETX> : FILE TERMINATION CODE

THIS CODE IS TRANSMITTED TO HOST SYSTEM WHEN A FILE RECORD TRANSMISSION IS LOST. AFTER THE HOST SYSTEM CLOSES THE FILE, IT TRANSMITS <ACK> AND TERMINATES THE OBJECT SAVE SEQUENCE.

<NAK> : SAVE SEQUENCE HALT INSTRUCTION CODE

THIS CODE INSTRUCTS THE ICD TO HALT THE OBJECT SAVE SEQUENCE. THE HALT CODE IS USED PRIMARILY FOR A TARGET MEMORY TIMEOUT OR NON-MEMORY ACCESS. ONCE THE FILE IS CLOSED THE HOST SYSTEM TRANSMITS <ACK>, HALTING THE OBJECT FILE SAVE SEQUENCE.

<ACK> : FILE CLOSE TERMINATION CODE
FILE CLOSE

THIS CODE IS SENT TO THE ICD WHEN A FILE IS ABLE TO BE CLOSED.

<NAK> : SAVE SEQUENCE HALT INSTRUCTION CODE

THIS CODE INSTRUCTS THE ICD TO HALT THE OBJECT SAVE SEQUENCE. IT IS USED WHEN THE HOST SYSTEM HAS DETECTED FILE READ, ICD AND HANDSHAKE ERRORS, ETC. IT IS POSSIBLE TO TRANSMIT <NAK> AND HALT THE SEQUENCE AT ANY TIME.

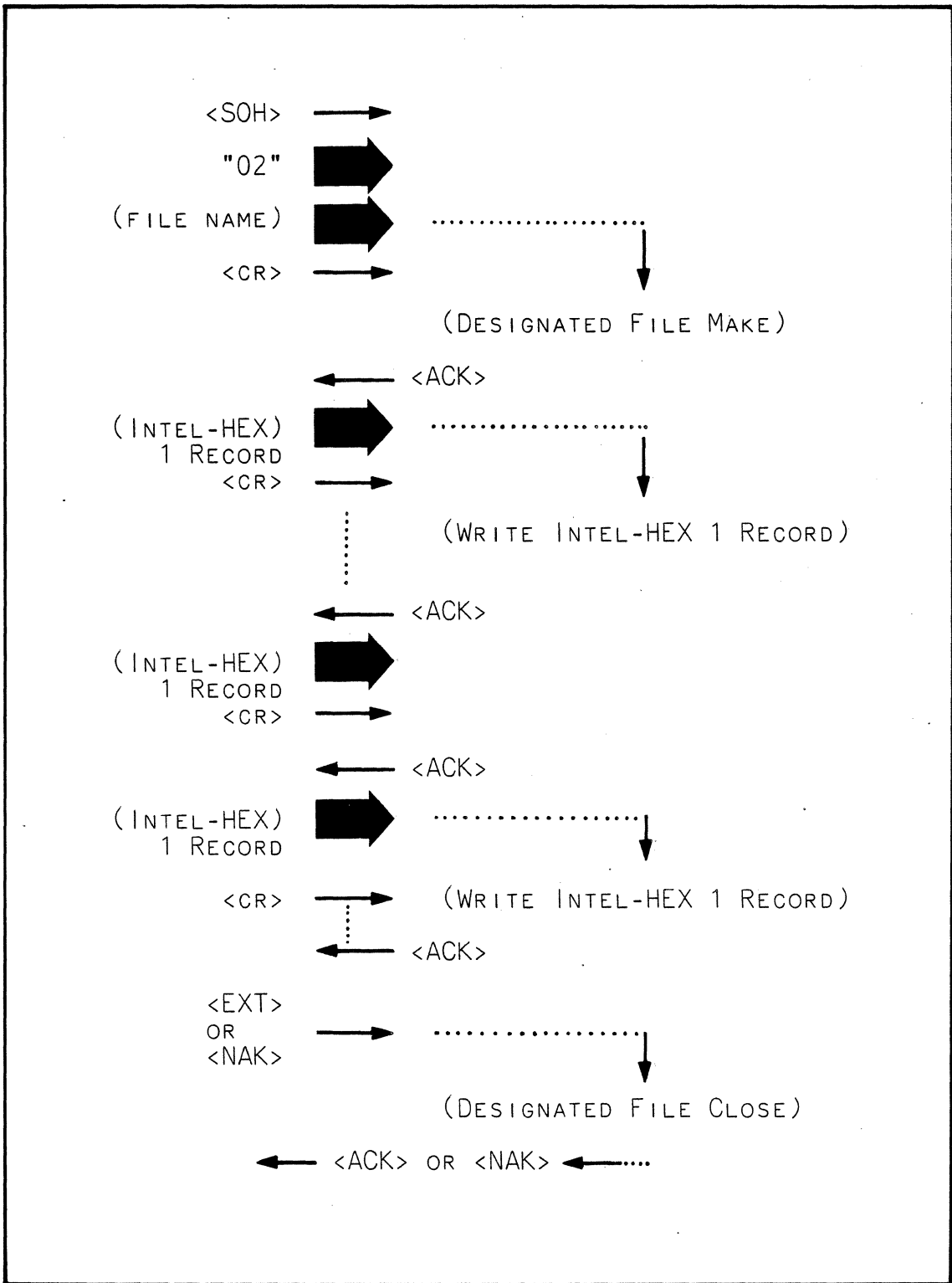


Figure 6-9. Object File Save Sequence

Z COMMAND

'Z' COMMAND SEQUENCE

THIS SEQUENCE EXECUTES THE 'Z' COMMAND PROCESS DEPENDING ON PARAMETERS SENT FROM THE ICD.

<SOH>"03"(PARAMETER)<CR> : 'Z' COMMAND RECORD

: THIS RECORD IS SENT FROM THE ICD TO ENABLE THE HOST SYSTEM TO EXECUTE THE 'Z' COMMAND PROCESS.

<ACK> : 'Z' COMMAND NORMAL TERMINATION CODE
THIS CODE IS SENT FROM THE HOST SYSTEM TO THE ICD WHEN THE 'Z' COMMAND PROCESS IS TERMINATED IN THE NORMAL MANNER. THE ICD IS UNAWARE OF THE 'Z' COMMAND PROCESS IN THE HOST SYSTEM.

<NAK> : 'Z' COMMAND ABNORMAL TERMINATION CODE
THIS CODE IS SENT FROM THE HOST SYSTEM TO THE ICD WHEN THE 'Z' COMMAND PROCESS IS TERMINATED IN AN ABNORMAL MANNER. THE ICD IS UNAWARE OF THE 'Z' COMMAND PROCESS IN THE HOST SYSTEM.

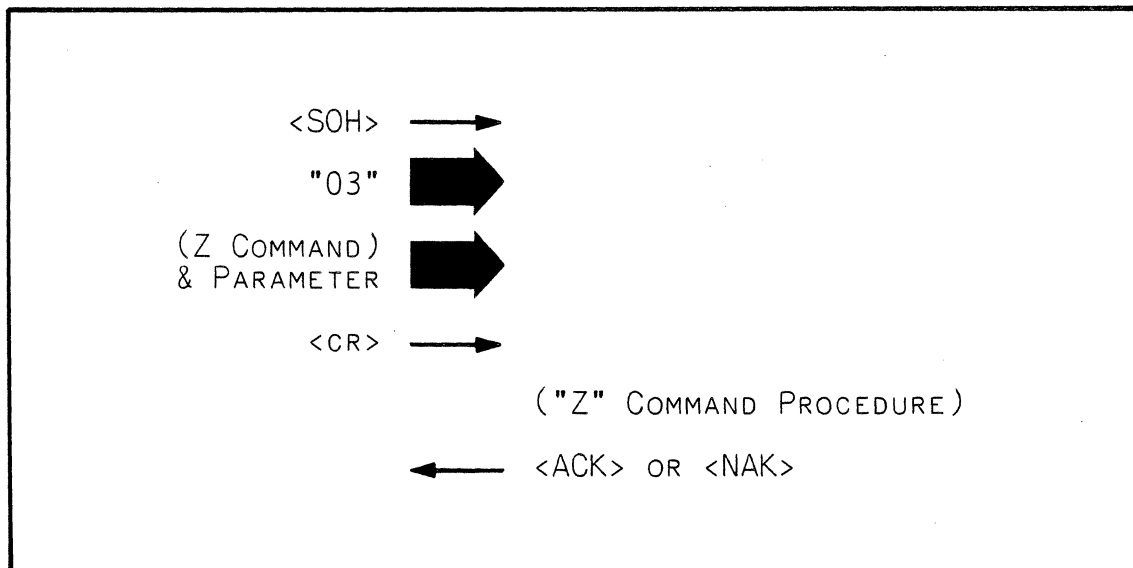


Figure 6-11. Z Command Sequence

'Z' COMMAND PROGRAM

- 1) THE ICD SENDS <SOH>"03"(PARAMETER)<CR> TO THE HOST SYSTEM BY THE 'Z' COMMAND.
- 2) IF <SOH>"03"(PARAMETER)<CR> IS RECEIVED, THE HOST SYSTEM WILL EXECUTE THE APPROPRIATE 'Z' COMMAND PROCESS.
 - A) IF AN ERROR OCCURS IN THE 'Z' COMMAND PROCESS, THE HOST SYSTEM WILL SEND <NAK> TO THE ICD AND THEN RETURN TO THE IDLE PROGRAM.
 - B) IF NO ERROR OCCURS IN THE 'Z' COMMAND PROCESS, THE HOST SYSTEM WILL SEND <ACK> TO THE ICD AND THEN RETURN TO THE IDLE PROGRAM.

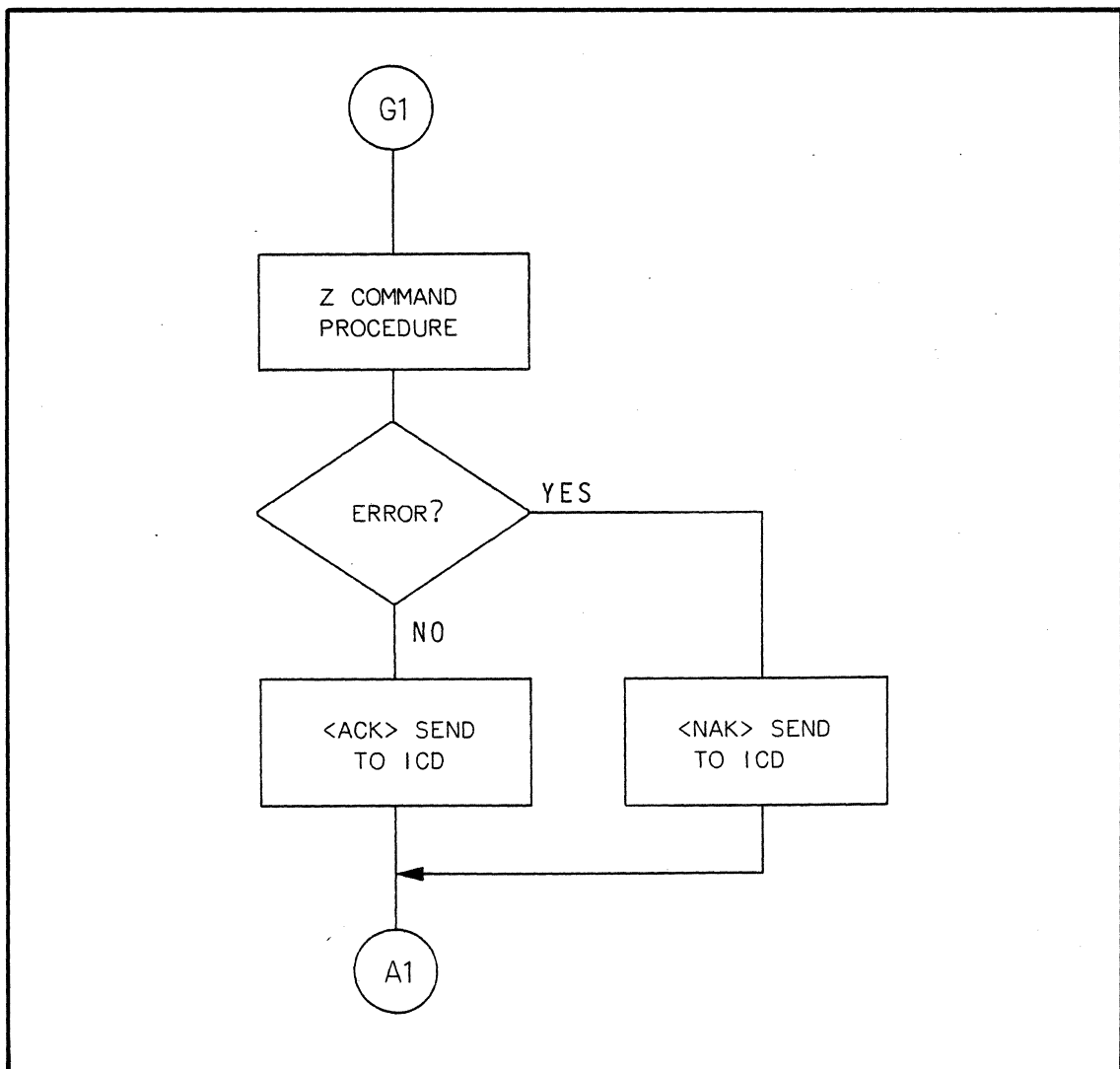


Figure 6-12. Z Command Flowchart

THIS PAGE INTENTIONALLY BLANK

QUIT

QUIT SEQUENCE

THIS SEQUENCE RETURNS THE HOST SYSTEM UNDER THE CONTROL OF THE OS VIA CODES SENT FROM THE ICD.

<SOH>"04"<CR> : QUIT RECORD
THIS RECORD IS SENT FROM THE ICD TO RETURN THE HOST SYSTEM UNDER THE CONTROL OF THE OS. THE ICD IS UNAWARE OF THE QUIT COMMAND PROCESS IN THE HOST SYSTEM.

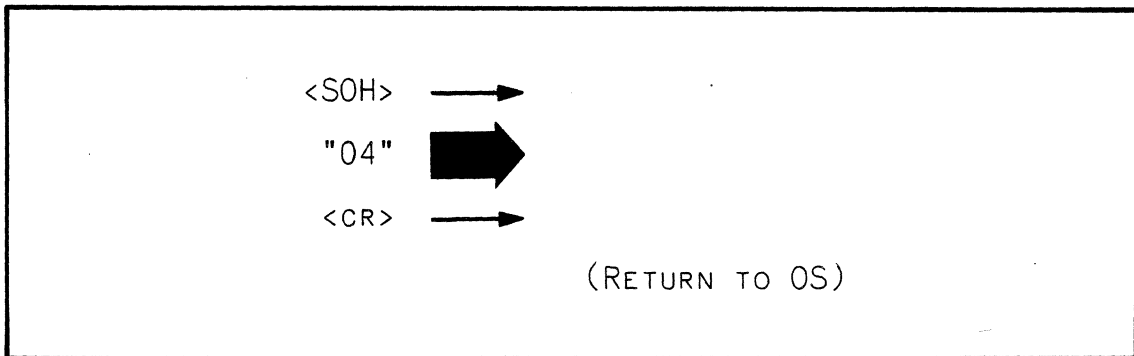


Figure 6-13. Quit Sequence

QUIT PROGRAM

- 1) THE ICD SENDS <SOH>"04"<CR> TO THE HOST SYSTEM WITH THE QUIT COMMAND .
- 2) IF <SOH>"04"<CR> IS RECEIVED, THE HOST SYSTEM RETURNS UNDER CONTROL OF THE OS.

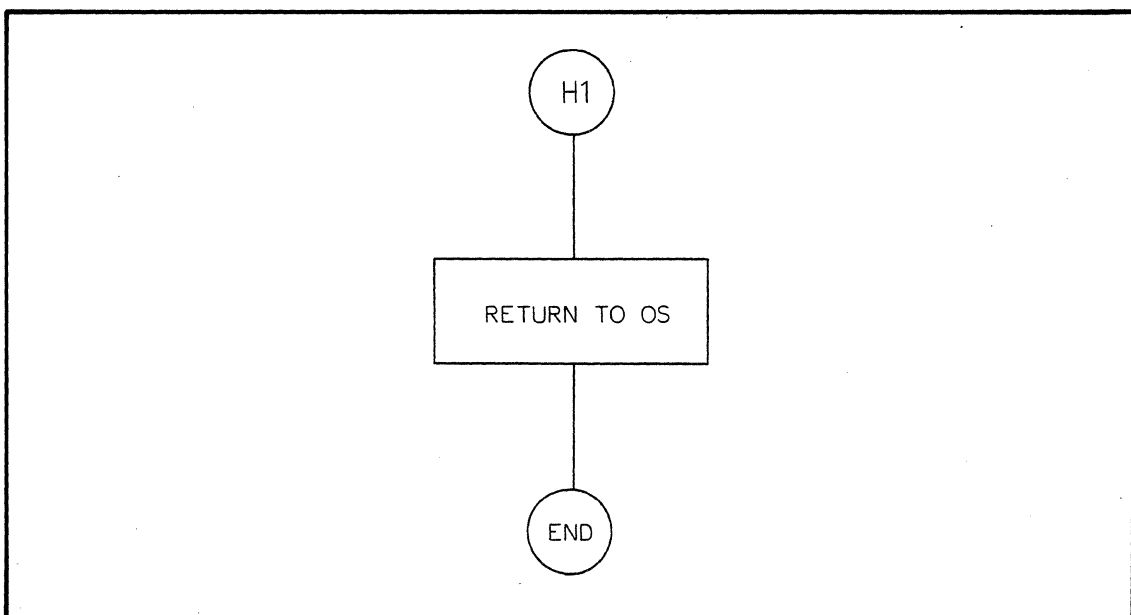


Figure 6-14. Quit Flowchart

DISPLAY SYMBOLIC TEXT

DISPLAY SYMBOLIC TEXT SEQUENCE

THIS SEQUENCE CHANGES A PARAMETER SENT FROM THE ICD TO A SYMBOL AND THEN DISPLAYS IT ON THE HOST SYSTEM CONSOLE.

...<SOH>"1x"(PARAMETER)...<CR> : SYMBOL CHANGE RECORD

: THIS RECORD IS SENT TO HOST SYSTEM WHEN ICD CHANGES A PARAMETER TO A SYMBOL. THE SYMBOLIC TEXT RECORDS, INCLUDING SYMBOL CHANGE RECORDS, ARE SIMILAR TO ASCII TEXT. MORE THAN ONE SYMBOL CHANGE CODE MAY BE INCLUDED IN A SYMBOLIC TEXT RECORD, EXCLUDING <ACK> OR <NAK> CONTROL CODES.

THE SYMBOL CHANGE RECORD PARAMETER, SPECIFIED AS "1x", ALLOWS VALUES BETWEEN "10" AND "1F" TO BE INSERTED. IF THE SYMBOLIC DISPLAY IS NOT PERFORMED, THE CHARACTERS (EXCEPT <SOH>) ARE DISPLAYED ON THE HOST SYSTEM CONSOLE.

- <ACK> : DISPLAY TERMINATION CODE
THIS CODE IS SENT FROM THE HOST SYSTEM TO THE ICD WHEN THE SYMBOL AND TEXT DISPLAY OF SYMBOLIC TEXT RECORD HAS BEEN DISPLAYED.
- <NAK> : DISPLAY HALT CODE
THIS CODE IS SENT FROM THE HOST SYSTEM TO THE ICD TO HALT THE TRANSMISSION OF SYMBOLIC TEXT RECORDS FROM THE ICD DURING THE DISPLAY OF "TRACE" ETC.

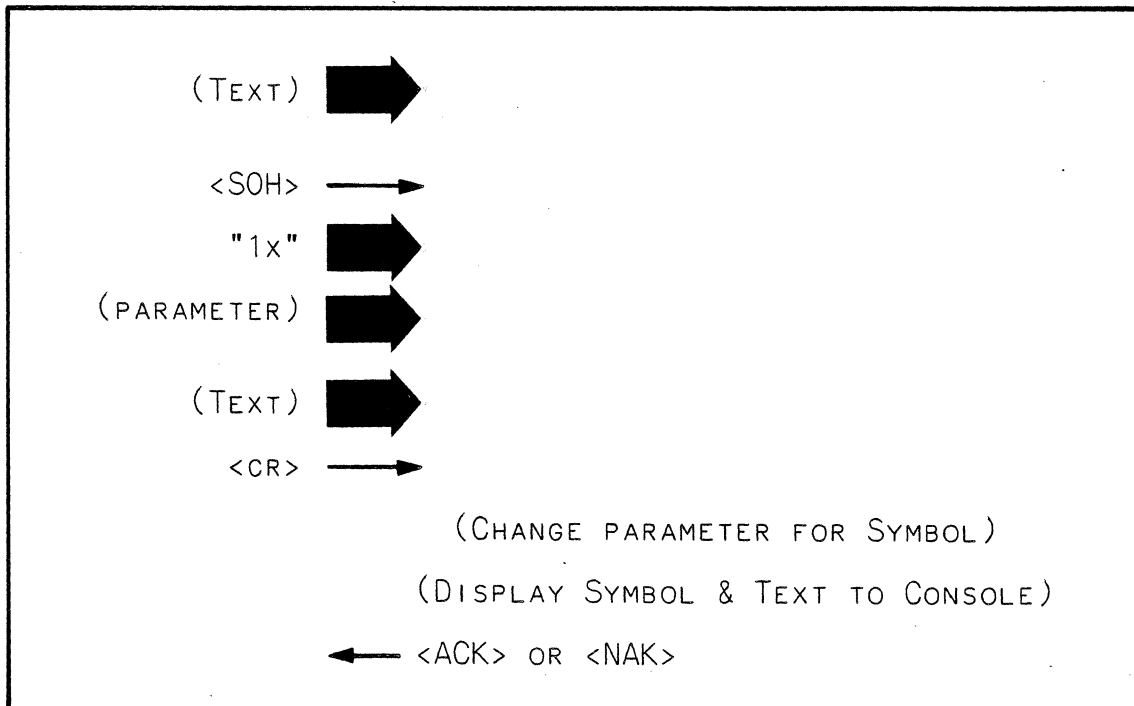


Figure 6-15. Display Symbolic Text Sequence

DISPLAY SYMBOLIC TEXT PROGRAM

- 1) THE ICD SENDS <SOH>"1X"(PARAMETER)<CR> TO THE HOST SYSTEM WHEN THERE ARE PARAMETERS THAT MUST BE REPRESENTED WITH SYMBOLS.
- 2) THE HOST SYSTEM ALLOWS BUFFERING OF ALL DATA UP TO <CR> IN THE ICD INPUT BUFFER.
- 3) THE HOST SYSTEM THEN SEARCHES THE INPUT BUFFER FOR <SOH>"1X"(PARAMETER);
 - A) IF <SOH>"1X"(PARAMETER) CANNOT BE FOUND, THE HOST SYSTEM WILL DISPLAY THE CONTENTS OF THE INPUT BUFFER (SYMBOLIC DISPLAY ON THE HOST SYSTEM CONSOLE), THEN SEND <ACK> TO THE ICD AND RETURN TO THE IDLE PROGRAM.
 - B) IF <SOH>"1X"(PARAMETER) CAN BE FOUND, THE HOST SYSTEM WILL SEARCH THE SYMBOL TABLE FOR THE PARAMETER.
 - 1) IF THE (PARAMETER) CANNOT BE FOUND IN THE SYMBOL TABLE, THE HOST SYSTEM WILL TRANSFORM <SOH>"1X"(PARAMETER) TO THE PARAMETER AND THEN RETURN TO STEP 3).
 - 2) IF THE (PARAMETER) CAN BE FOUND IN THE SYMBOL TABLE, THE HOST SYSTEM WILL TRANSFORM <SOH>"1X"(PARAMETER) IN THE INPUT BUFFER, TO THE SYMBOL AND THEN RETURN TO STEP 3).

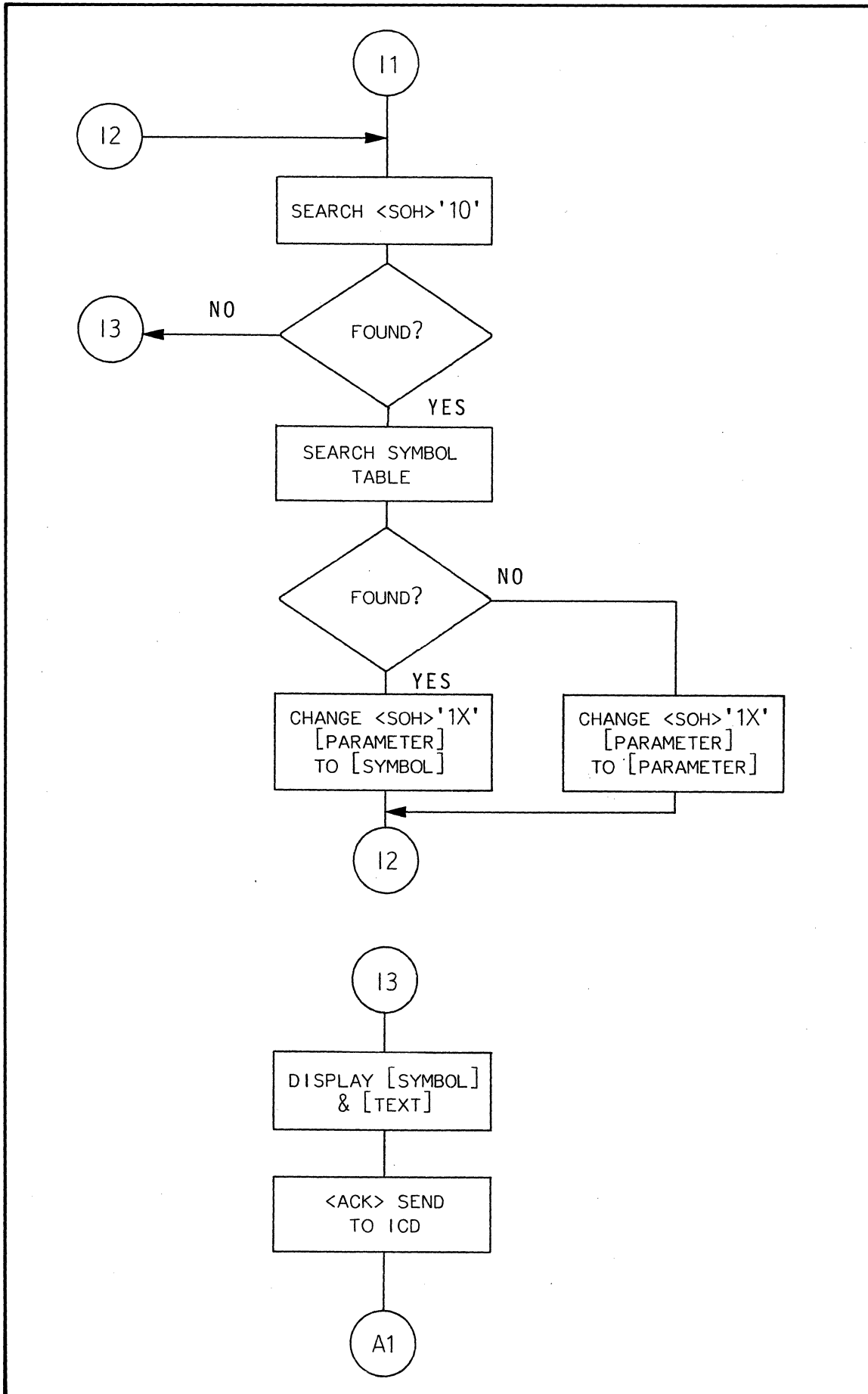


Figure 6-16. Display Symbolic Text Flowchart

CONSOLE KEY CHECK PROGRAM

CONSOLE KEY CHECK SEQUENCE

THIS SEQUENCE ENABLES THE ICD TO CHECK THE HOST SYSTEM CONSOLE INPUT FOR THE SUSPENSION AND RESUMPTION OF TRACE OR DUMP OUTPUT AND HALT INQUIRY.

<SOH>"09"<CR> : CONSOLE KEY INPUT CHECK REQUEST

THE ICD REQUESTS THE CONSOLE KEY INPUT CHECK IN REGARD TO THE HOST SYSTEM.

<ACK> : CONSOLE CODE MINUS INPUT

THIS CODE IS TRANSMITTED TO THE ICD WHENEVER THERE IS NO CONSOLE KEY INPUT.

<ANY ASCII CODE>: CONSOLE INPUT CODE

WHEN A CONSOLE KEY INPUT OCCURS, THE HOST SYSTEM TRANSMITS ONLY THE 1 CODE OF THE ASCII DATA TO THE ICD.

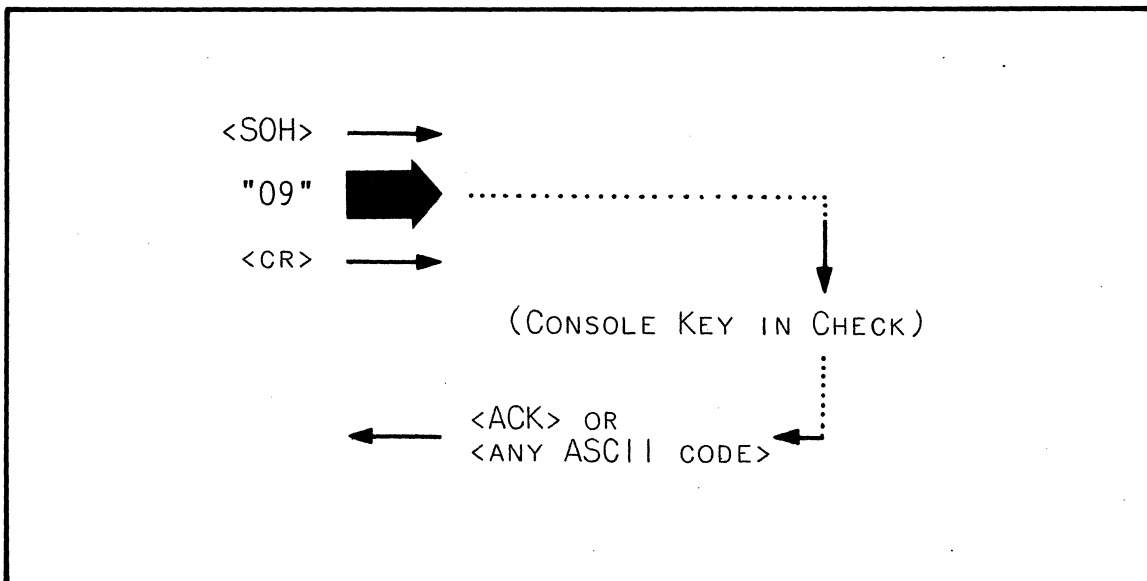


Figure 6-17. Console Key Check Sequence

POWER/GROUNDING REQUIREMENTS

Power Requirements : 100-120VAC \pm 10% (100V - 117V)
 200-240VAC \pm 10% (200V - 240V)

Fuse : 2A

POWER

The ICD-278 can be used with a single phase power supply of 100 - 240 VAC by selecting the proper switch settings. The ICD-278 uses the same primary supply as the target system. This power supply contains filters that reduce the AC line noise.

The JVCC jumper is used to connect the target system and the +5V line of the ICD. In most cases, this jumper should be removed. The ICD logic GND is connected with the Terminal, SG.FG of the Host/Aux and the twisted pair shield of the in-circuit probe.

GROUND

The GND of the target system, twisted pair shield and the ICD ground, are connected through the plug probe. This arrangement is necessary for precise emulation quality.

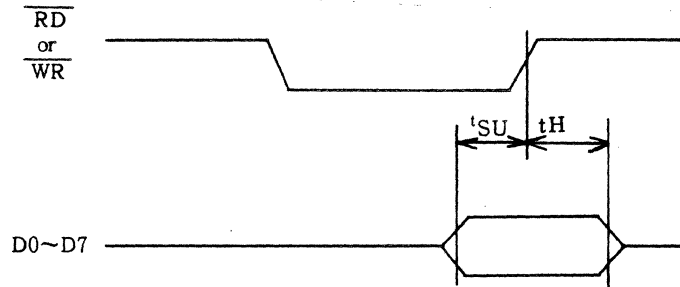
When configuring an ICD development system, be aware of the following;

- 1) Use the same primary power source to power the ICD, target system, console terminal, host computer system etc.
- 2) Common frame grounding will improve the system performance.
- 3) The in-circuit probe may pick up noise from electromagnetic devices such as relays and motors. If this becomes a problem, additional shielding of the in-circuit probes may be necessary.

REALTIME TRACE ACQUISITION

During realtime tracing, storage of the data bus signals occur on the leading edge of the RD, WR, M1 or IORQ signals.

The ICD-2178 ensures positive storing of the address, status and data bus signals while emulating with ICD program memory. However, read data cannot be stored when the data setup time is shorter than 90ns.



t_{SU}	Real Time Trace User Data Read set up time	min 90 ns
t_{H}	Real Time Trace User Data Read hold time	min 0 ns

ZAX

TECHNICAL BULLETIN

200 -08

DATE: 12/14/83

TO: All ICD 178-8086/88 and 278 users

SUBJECT: Communication Protocol Verification Method

The following procedure will aid the ICD user in understanding the correct communication protocol (object upload/download) between the Emulator and the Host Computer. The example shown may be used in conjunction with Sequence Flow Charts located in the Software Specification section of the Operation Manual.

The example discussed utilizes two Televideo Model 925 terminals and one ZAX ICD-178 8086/88.

- 1) Configure the Figure 1;

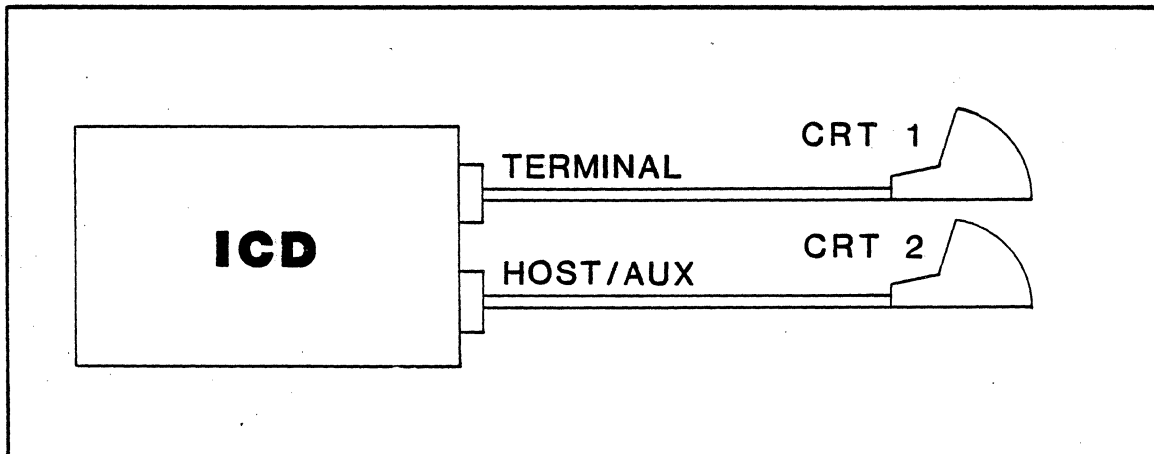


Figure 1

- 2) Set the DCE/DTE ICD switch to - DCE.
- 3) Set the baud rate of the ICD to - 9600bps.
- 4) Set the Local/Remote switch to - LOCAL.

NOTE: CRT 1 is used as a normal terminal. CRT 2 is used as a host computer, that is, it is used to receive and send data to the ICD. For this operation, CRT 2 must be set in the Half Duplex and Monitor mode. The Half Duplex mode is selected by setting pins 7 & 8 of switch S2 (Function) to the bottom (down) position (Figure 2).

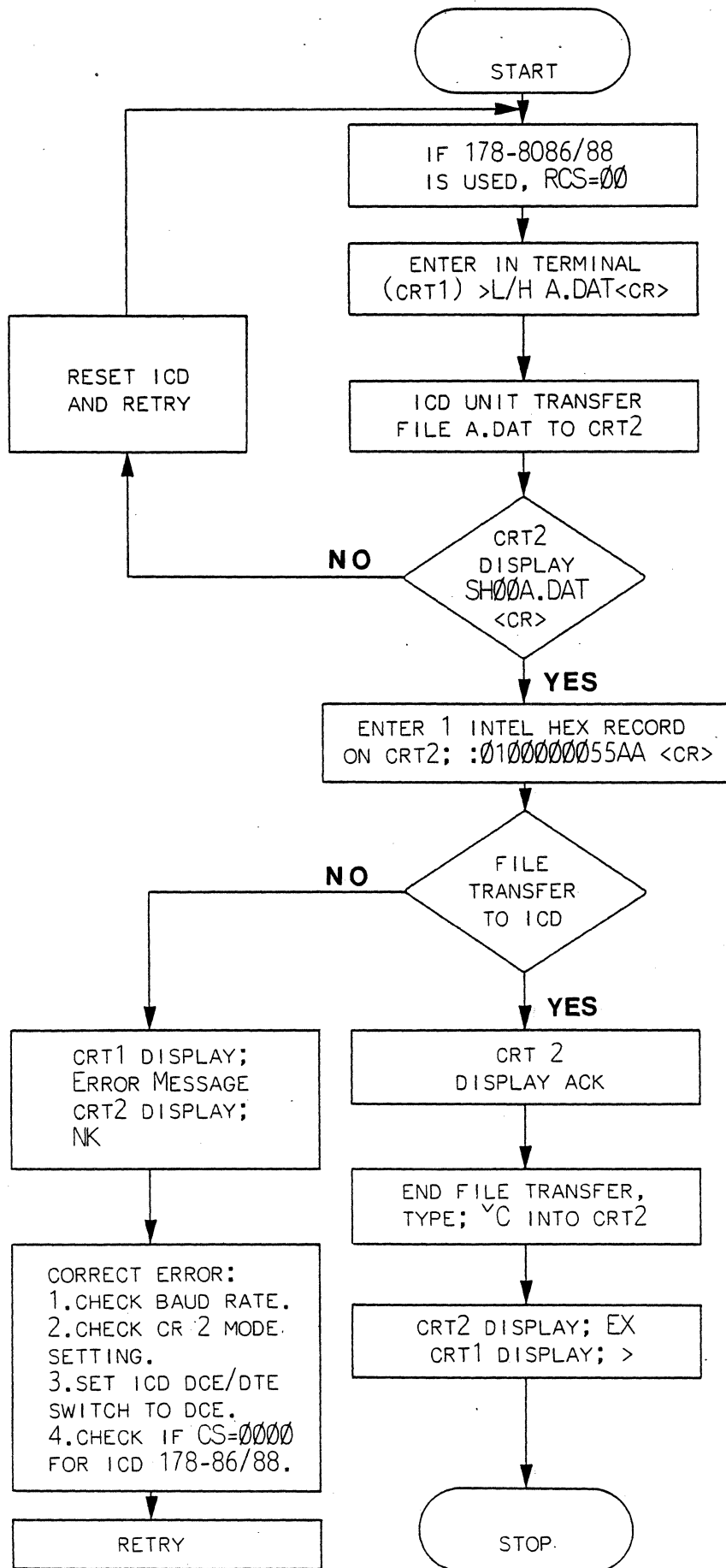
	1	2	3	4	5	6	7	8	9	10
UP						X			X	
DOWN	X	X	X	X	X		X	X		X

Figure 2

The Monitor mode setting allows display of all characters received from the computer or entered into the keyboard, including escape and control sequences (The escape and control sequences provide visual cues during long program routines). The Monitor mode is accomplished by entering "ESC U <cr>" to the keyboard. Entering "ESC X <cr>," will return the terminal back to normal operation.

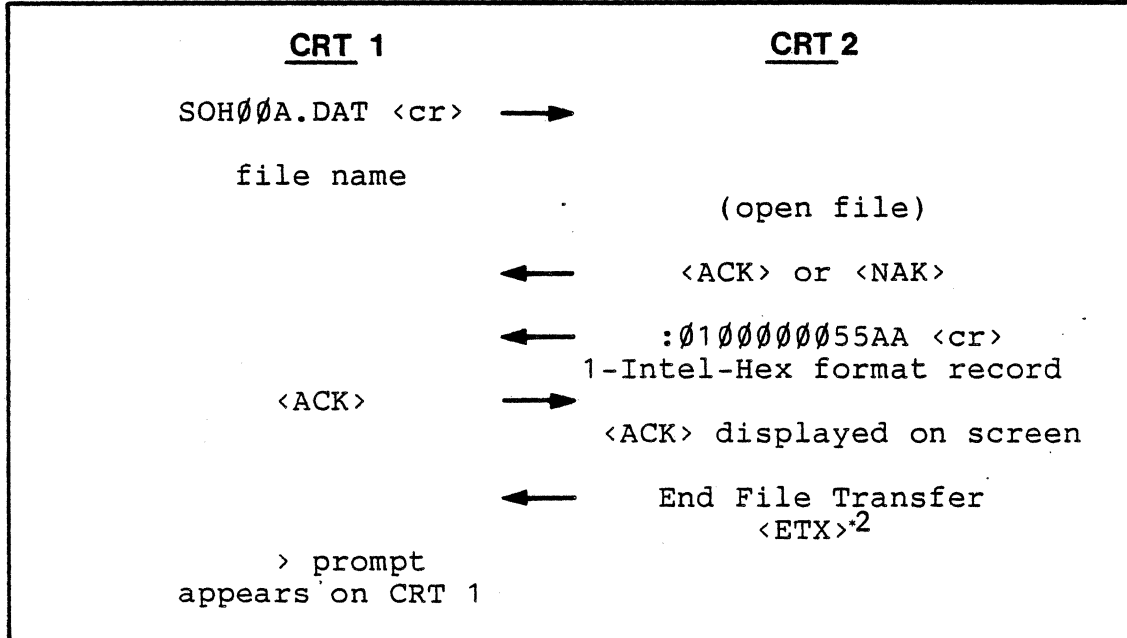
OPERATION PROCEDURE

The verification operation procedure of this example will be illustrated by both, Flow Chart and Sequence Diagram.



SEQUENCE DIAGRAM

Object File Load/Verify Sequence



NOTE 1) If ICD model 178 8086/88 is used, the code segment must be set to Ø by the "Register" command; ">RCS=Ø".

NOTE 2) Control character "ETX" is generated by entering in "(control) C" on CRT 2.

NOTE 3) VT52 Terminal or equivalent, will display the following letters;

Start of Heading	- SOH	= A
End of Text	- ETX	= C
Acknowledge	- ACK	= F
Carriage Return	- cr	= M



TECHNICAL BULLETIN

200-07

DATE: 12/15/83

TO: All ICD 278/Z80 & 278/8085 Users

SUBJECT: RANGE BREAKING

ITEM ONE:

The following examples display the ICD 278/Z80's Range Breaking Conditions using different combinations of the 3 available hardware breakpoints. Two examples are shown.

NOTE: In Examples 1 and 2, Addresses are divisible by 2 and 16-bit binary address is used for Range Breaking.

EXP.1 <<Range Break Address A=0020H,0021H,0030H,0031H>>
<<Range Break Address B=0040H,0060H>>

```
>I0
>F 0,FFFF,0 (fills emulator memory with NOP)
>B/A OF,0000_0000_001X_000X
>B/B OF,0000_0000_01X0_0000
>B
  A(ON)OF00XX (0000_0000_001X_000X)
  B(ON)OF00X0 (0000_0000_01X0_0000)
>G0
  PC
  0020H NOP
  <Break Hardware A>
>G
  PC
  0021H NOP
  <Break Hardware A>
>G
  PC
  0030H NOP
  <Break Hardware A>
>G
  PC
  0031H NOP
  <Break Hardware A>
>G
  PC
  0040H NOP
  <Break Hardware B>
>G
  PC
  0060H NOP
  <Break Hardware B>
```

End EXAMPLE 1



TECHNICAL BULLETIN

EXP.2 <<Break Between Addresses 1000H and 7FFFH>>
<<Break Range A=1000H to 1FFF & 3000H-7FFFH>>
<<Break Range B=2000H to 3000H>>

>F,0,FFFF,0 (used to fill emulator with NOP)
>B/A OF,0XX1_XXXX_XXXX_XXXX
>B/B OF,0010_XXXX_XXXX_XXXX

>B
A(ON) XXXX 0XX1_XXXX_XXXX_XXXX
B(ON) 2XXX 001X_XXXX_XXXX_XXXX

>A 1000
1000 JP 1FFFH #
1003 <cr>

>A 1FFF NOP
2000 JP 2FFFH #
2003 <cr>

>
>A 2FFF NOP
3000 JP 7FFFH #
3003 <cr>

>G0
PC
1000 JP 1FFFH #
<Break Hardware A>

>G
PC
1FFF NOP
<Break Hardware A>

>G
PC
2000 JP 7FFFH
<Break Hardware B>

>G
PC NOP
2FFF
<Break Hardware B>*

>G
PC
3000 JP7FFFH
<Break Hardware A>

*NOTE: Both A & B breakpoints are enabled

#NOTE: NOP can be used in place of JMP and JP instructions but must use different >G (Address) to observe various range break addresses.

NOTE: For 278/8085 use JMP in place of JP instruction

End EXAMPLE 2



TECHNICAL BULLETIN

ITEM TWO:

Below is the procedure for using an address range NOT divisible by 2.

1. Specify H/W or S/W breakpoint at beginning of range.
2. Execute program until breakpoint occurs.
3. Set 2 additional breakpoints;
 - A. H/W or S/W breakpoint for ending address of break range.
 - B. H/W breakpoint using "don't care" conditions to cover entire address range.
4. Execute program from current location.
NOTE: Breakpoint should occur between specified address range.



Zax Corporation 2572 White Road, Irvine, California 92714
(714) 474-1170 • 800-421-0982 • TLX 183829