INTRODUCTION

The information contained in this COMMAND REFERENCE GUIDE relates to emulators directed at Z-80B based devices.

The ICD-278 is capable of initiating 29 different software commands. Each command is structured in a particular format that must be selected and executed accurately. It is therefore important for the design engineer to become familiar with the proper command instruction and results in order to use the emulator commands effectively.

For easy identification and immediate usage, symbolic notation will be used to explain the proper input format. This notation will be in the form of a console keyboard illustration, typical of the type an operator would be using.

Each console keyboard element will symbolize the EXACT input operation which must be performed, including spacing, feature keys, optional characters, numbers, etc. The operator has only to determine which command format to use and then follow the format command configuration displayed for each command.

Each command is defined by its symbolic key letter, the command name, an explanation and an example of typical command usage.
<table>
<thead>
<tr>
<th>TABLE of CONTENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address/Data parameters .......... page 4</td>
</tr>
<tr>
<td>Symbol Notation ................. 5</td>
</tr>
<tr>
<td>Command Example Format .......... 6</td>
</tr>
</tbody>
</table>

**COMMANDS:**

<table>
<thead>
<tr>
<th>Command</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASSEMBLE</td>
<td>8</td>
</tr>
<tr>
<td>BREAK</td>
<td>10</td>
</tr>
<tr>
<td>CALCULATION</td>
<td>18</td>
</tr>
<tr>
<td>COMPARE</td>
<td>20</td>
</tr>
<tr>
<td>DUMP</td>
<td>22</td>
</tr>
<tr>
<td>DISASSEMBLE</td>
<td>24</td>
</tr>
<tr>
<td>EXAMINE</td>
<td>26</td>
</tr>
<tr>
<td>EVENT</td>
<td>28</td>
</tr>
<tr>
<td>FILL</td>
<td>30</td>
</tr>
<tr>
<td>GO</td>
<td>32</td>
</tr>
<tr>
<td>HISTORY</td>
<td>34</td>
</tr>
<tr>
<td>INCIRCUIT</td>
<td>38</td>
</tr>
<tr>
<td>IDENTIFICATION</td>
<td>40</td>
</tr>
<tr>
<td>LOAD</td>
<td>42</td>
</tr>
<tr>
<td>MOVE</td>
<td>44</td>
</tr>
<tr>
<td>MAP</td>
<td>46</td>
</tr>
<tr>
<td>NEXT</td>
<td>48</td>
</tr>
<tr>
<td>OFFSET</td>
<td>50</td>
</tr>
<tr>
<td>PORT</td>
<td>52</td>
</tr>
<tr>
<td>PIN</td>
<td>54</td>
</tr>
<tr>
<td>PRINT</td>
<td>56</td>
</tr>
<tr>
<td>QUIT</td>
<td>58</td>
</tr>
<tr>
<td>REGISTER</td>
<td>60</td>
</tr>
<tr>
<td>SEARCH</td>
<td>62</td>
</tr>
<tr>
<td>SAVE</td>
<td>64</td>
</tr>
<tr>
<td>SUPERVISOR</td>
<td>66</td>
</tr>
<tr>
<td>TRACE</td>
<td>68</td>
</tr>
<tr>
<td>USER</td>
<td>70</td>
</tr>
<tr>
<td>VERIFY</td>
<td>72</td>
</tr>
<tr>
<td>Error Messages</td>
<td>74</td>
</tr>
<tr>
<td>Appendix A</td>
<td>76</td>
</tr>
<tr>
<td>B</td>
<td>80</td>
</tr>
<tr>
<td>C</td>
<td>82</td>
</tr>
<tr>
<td>D</td>
<td>84</td>
</tr>
</tbody>
</table>
ADDRESS/DATA Parameters

Throughout this guide, the majority of commands will require the user to input address or data information. This section describes the address/data parameters available and explains their usage. Also displayed are the various symbol notations and their explanation.

The address/data common to the ICD commands is displayed to the right of the parameter explanations and characterized by the symbol: 

This parameter represents a hexadecimal PHYSICAL memory address in hex 16-bit. It can be specified by offset registers, addition or subtraction. In a "DON'T CARE" situation, "X" may be specified for the BREAK and EVENT commands (in this case, the parameter cannot be specified by the offset registers, addition or subtraction).

This parameter represents a HEXADECIMAL 16-bit memory address.

This parameter represents a BINARY 16-bit physical memory address. It can be used ONLY for a hardware break address in the BREAK and EVENT commands, and is displayed whenever using the BREAK status command.

This parameter specifies data modification in ASCII codes. All ASCII A thru Z characters (32) may be used.

This parameter specifies to use WORD or BYTE data (8/16-bit). The valid range is ø to F and "X" (don't care).

This parameter is used to specify the number of BYTES. The proper format for data entry is;

"L" (number of bytes)

NOTE: The letter "L" must be specified before the number can be entered.
SYMBOL Notation

This symbol is used to represent a keyboard character that the operator will utilize. NOTE: Consult your console keyboard operating manual for validation of characters.

This symbol displays to the operator that an address, data group, letter code, etc., is needed at this point in the command format. The options are always explained immediately after this symbol or in the address/data box.

This symbol is used to represent a response from the system via the console display. The symbol will visualize all data exactly as it appears on the console screen.

This symbol indicates additional information may be found in the APPENDIX section.

This symbol locates various notes throughout the Command Reference Guide.
In this example, "M" is the symbol used to initiate a MOVE command. Since the letter "M" is used to specify the MOVE command, there are no options available yet.

1. This specifies a data entry that is requested. The available addresses will always be displayed just to the right of the command format explanation. 

2. This specifies a second data entry. In this example, the memory address at which the data transfer is terminated. The available addresses to the user are.

3. This specifies a third data entry. In this example, a beginning memory address for the data transfer. The available address to the user is.

4. This specifies the option of using either UP or PU to determine how data is to be transferred between the ICD and the target system.

PU, UP < The available options for the keyboard characters will always be displayed to the left of the explanation.

EXAMPLE

Move the data that resides in the Target System memory \( \text{\$0000} \) - \( \text{\$0FFF} \), to the Emulator memory. Start at the ICD address, \( \text{\$0000} \);

\[ \text{M \$0000, \$0FFF, \$0000, UP} \]
**EXAMPLE**

**IN LINE ASSEMBLER:**

```
> A 100
0100 LD HL, A000H
0103 PUSH DE
0104 LD DE, B000H
0107 EX DE, HL
0108 POP DE
0109 INC HL
010A INC DE
010B JP 0100H
010E
```

The ASSEMBLE command is used for writing software patches employing the internal 64K memory of the ICD emulator.

1. Specifies the memory address
2. Console displays the ICD address
3. Specifies a mnemonic code to be used in the Z80 assembler
HARDWARE Break
Type 1: Standard Hardware Break;

1: A, B, C specifies a hardware break name.
   (If none is specified, assume A then B then C after
   which occurs the error message; BREAK BUSY)

2: Specifies the break status:
   M Memory access
   M R Memory read
   M W Memory write
   P Port access
   P R Port read
   P W Port write
   O F Op code fetch
   I A Interrupt acknowledge

3: Specifies a break address PHY 16

4: Specifies the break passing count
   (Decimal 1 to 65535; Default=1)
HARDWARE Break
Type 2 Event Trigger Break;

1: A, B, C specifies a hardware break name
   (see Type 1; Standard Hardware Break)

2 specifies an arm/individual break;

   ARM the ARM break occurs only after the event
   trigger takes place.

   IND the individual break takes place without
   regard to the event trigger.

The BREAK command is used to set the standard hardware and software
breakpoints. Supplemental breakpoint commands include setting an
event trigger break, an eternal break, an operational break and
displaying a break condition. Supplemental commands may be used in
conjunction with the standard hardware and software break commands.
> B / OPT 1 cr

- HARDWARE Break
  Type 3 External Break;

OPT: X
  SPECIFIES AN OPTIONAL EXTERNAL BREAK

1:
  SPECIFIES THE EXTERNAL BREAK TO OCCUR AT
  THE HI OR LOW EDGE

H I , L O W

> B / 1 2 cr

- HARDWARE Break
  Type 4 Break Condition;

1: A, B, C
  SPECIFIES A HARDWARE BREAK NAME
  (SEE TYPE 1; STANDARD HARDWARE BREAK)

2
  MAKES A SPECIFIED BREAK:

  O N
  Valid

  O F F
  Invalid

  C L R
  Cleared

- NOTE: IF THE SPECIFIC HARDWARE BREAK NAME IS OMITTED,
  ALL HARDWARE BREAKS ARE AFFECTED.
SOFTWARE Break
Type 1 Standard Software Break:

1: 0-7 SPECIFIES A SOFTWARE BREAK NAME
   (MUST SPECIFY)
2 SPECIFIES A BREAK ADDRESS
3 SPECIFIES A BREAK PASSING COUNT
   (DECIMAL 1 TO 65535; DEFAULT=1)

SOFTWARE Break
Type 2 Enable/Disable Break:

1: SPECIFIES TO PLACE SOFTWARE BREAK IN THE ENABLE STATE
2: SPECIFIES TO PLACE SOFTWARE BREAK IN THE DISABLE STATE
   (NO SOFTWARE BREAK OCCURS)

NOTE: WHEN THE SOFTWARE BREAK IS IN THE ENABLE STATE,
execute MOV A,A (7FH); AND A BREAK WILL OCCUR.
Breaks can be set using an arbitrary number by
generating them in the user program.
SOFTWARE Break
  Type 3 Break Condition:

1: O-7 SPECIFIES A SOFTWARE BREAK NAME  
   (MUST SPECIFY)

2: MAKES A SPECIFIED SOFTWARE BREAK:
   
   ON  VALID
   OFF INVALID
   CLR CLEARED

1 NOTE: IF THE SPECIFIC SOFTWARE BREAK NAME IS OMITTED, 
       ALL SOFTWARE BREAKS ARE AFFECTED.

SUPPLEMENTAL Break
  Type 1

1: T SPECIFIES A WAIT TIMEOUT
   E SPECIFIES AN EVENT TRIGGER
   W SPECIFIES A WRITE PROTECT

2 MAKES ANY OF THE 1 COMMANDS:
   ON  VALID
   OFF INVALID
SUPPLEMENTAL Break
Type 2

1: E
SPECIFIES AN EVENT TRIGGER BREAK

2
SPECIFIES A BREAK PASSING COUNT
(DECIMAL 1 TO 65535; DEFAULT=1)

SUPPLEMENTAL Break
Type 3

1: I N I
CLEARS THE EVENT PASSING CONDITION BY resetting
the condition of the ARM occurrence
EXAMPLE: 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 Ø100 1 0 IND (0001_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_XXX0_0000
>B
A (ON) M XXX0 1 0 IND (000X_111X_XXX0_0000)
T (ON)
S (DI)
W (OFF)
EXAMPLE: BREAK Command

Set a software break using a hexadecimal address:

```
> B/7 1234
> B
A (ON) M XXXØ 1 Ø IND (000X_111X_XXXX_0000)
7 (ON) 1234 1 Ø
T (ON)
S (DI)
W (OFF)
```

Set arm break for hardware break A:

```
> B/A ARM
> B
A (ON) M XXXØ 1 Ø ARM (000X_111X_XXXX_0000)
7 (ON) 1234
T (ON)
S (DI)
W (OFF)
```

Enable a software break:

```
> B S=EN
> B
A (ON) M XXXØ 1 Ø ARM (000X_111X_XXXX_0000)
7 (ON) 1234
T (ON)
S (EN)
W (OFF)
```
EXAMPLE

Execute multiple handling of subtraction and addition using hexadecimal and decimal data interchangably;

\[ >C \text{ 1234+1234} \]
H: 0009A4
D: 2468

\[ >C \text{ 1000H-FFFH} \]
H: 000001
D: 1

\[ >C \text{ 1} + \text{2} + \text{0} \cr \]

1 - 2 - 0 specifies the operation data

NOTE: Decimal is valid for -8388608 to +8388607
Hexadecimal valid for 0 to FFFFFF(H)

The calculation command is used to add and subtract hexadecimal and decimal numbers. The operation of addition and subtraction on data may be performed together on the same line.
1. Specifies the comparison start memory address (PHY 18).
2. Specifies the end memory address (PHY 18, L BYTE).
3. Specifies the start memory address to be compared (HEX ADDR).
4. Specifies either:
   - U P to use 1 for the user memory, and 3 for the ICD program memory,
   - P U to use 1 for the ICD program memory and 3 for the user memory.

**NOTE:** U P is used to compare target system memory contents to emulator memory contents.

**NOTE:** P U is used to compare emulator memory contents to target system memory contents.

**EXAMPLE**

Memory comparison of the range; 0 to 0001

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

<table>
<thead>
<tr>
<th>Address</th>
<th>1</th>
<th>Address</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>M</td>
<td>0000</td>
<td>M</td>
</tr>
<tr>
<td>C3</td>
<td></td>
<td>0000</td>
<td>FF</td>
</tr>
<tr>
<td>0001</td>
<td>06</td>
<td>0001</td>
<td>8F</td>
</tr>
</tbody>
</table>

**NOTE:**

1. Target Sys Memory Address
2. Target Sys Memory Contents
3. Emulator Memory Address
4. Emulator Memory Contents

**COMPARE**

The COMPARE command is used to compare the contents of specified memories and then display the unmatching data.
### Example

Dump memory contents from 0 to 30:

\[
> \text{D } 0, 30
\]

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>ASCII CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>21</td>
<td>FF</td>
<td>00</td>
<td>11</td>
<td>34</td>
<td>12</td>
<td>77</td>
<td>12</td>
<td>13</td>
<td>23</td>
<td>B7</td>
<td>C2</td>
<td>06</td>
<td>00</td>
<td>C3</td>
<td>00</td>
</tr>
<tr>
<td>0010</td>
<td>01</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0020</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0030</td>
<td>00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Dump memory contents into word:

\[
> \text{D/W } A000, L40
\]

<table>
<thead>
<tr>
<th>0</th>
<th>2</th>
<th>4</th>
<th>6</th>
<th>8</th>
<th>A</th>
<th>C</th>
<th>E</th>
<th>ASCII CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>A000</td>
<td>F07F</td>
<td>A18F</td>
<td>F005</td>
<td>900F</td>
<td>F01D</td>
<td>AFAF</td>
<td>F00D</td>
<td>001C</td>
</tr>
<tr>
<td>A010</td>
<td>F017</td>
<td>E5AF</td>
<td>F005</td>
<td>240F</td>
<td>F005</td>
<td>BDAF</td>
<td>F0ED</td>
<td>180F</td>
</tr>
<tr>
<td>A020</td>
<td>D00</td>
<td>84AF</td>
<td>F009</td>
<td>010F</td>
<td>D21F</td>
<td>A74F</td>
<td>F025</td>
<td>021F</td>
</tr>
<tr>
<td>A030</td>
<td>F00D</td>
<td>A125</td>
<td>F005</td>
<td>000F</td>
<td>F30F</td>
<td>A7AF</td>
<td>F045</td>
<td>520F</td>
</tr>
</tbody>
</table>
The DUMP command is used to display the memory contents in either BYTE or WORD units. The contents is represented by a hexadecimal number and ASCII code.
**DI**

**DI** SPECIFIES THE STARTING MEMORY ADDRESS (IF OMITTED, DATA IS DISPLAYED FROM THE CURRENT PC)

**EXAMPLE**

```
> DI 100
0100 210000 LD HL, B000H
0103 1100A0 LD DE, AAAAAH
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 210000 LD HL, B000H
0103 1100A0 LD DE, AAAAAH
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)
```

**NOTE:** >DI <cr>, DISPLAYS 12 LINES OF CODE.

**NOTE:** IF **1** AND **2** ARE OMITTED, THE DATA DISPLAYED WILL BE THE CURRENT PROGRAM COUNTER.

**DI** THE DISASSEMBLE COMMAND IS USED TO DISASSEMBLE THE MEMORY CONTENTS AND DISPLAY THE RESULT.
**Type 1**: Examine and Change memory contents

**Type 2**: Examine memory contents only

1. `/ W` specifies to change the memory contents on a word basis (default is a byte basis).
2. `3` specifies beginning address of memory to be filled.
3. `4` specifies the memory conversion data/conversion starting memory address.

- **Note**: Multi-display of data is possible using the format listed below:
  - `cr` displays data at the next address after change.
  - `cr` displays data at the same address after change.
  - `cr` displays data at the address that is decremented by 1 after the change.
  - `/ cr` terminates the examine command after the change.

**Example**

Examine and change memory in word data:

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

Examine and change memory in byte data:

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

**Note**: If `@` is omitted, the memory contents are not converted.

The EXAMINE command is used to change the memory contents to either ASCII or hexadecimal data. Two types of EXAMINE commands are available to the user: (1) Examine and change memory contents, (2) Examine the memory contents only.
**Standard Event Trigger**

**Type 1:**

1. **S T** specifies an event trigger status by name:
   - **M** Memory Access
   - **P W** Port Write
   - **M R** Memory Read
   - **O F** Op Fetch Code
   - **M W** Memory Write
   - **I A** Interrupt Acknowledge
   - **P** Port Access
   - **E X** Command Execution
   - **P R** Port Read

2. **A** specifies to direct an event trigger at a particular break address

3. **D** specifies event data (byte data)

4. If byte data is used, specifies the data

   **Bin-16**
   **Hex-Data**

   "XXX"
The EVENT command is used to set or release an event trigger. The standard event trigger is specified by a status name, address and event data. It is also possible to output an event trigger pulse via the event trigger connector during a specified machine cycle (type 2). An event trigger can also be used as a hardware break.

Type 2:

0 specifies to:

- Make an event trigger enabled
- Make an event trigger disabled
- Clear an event trigger

1 Note: "X" may be specified as "don't care" for binary or hexadecimal input (1 or 4 bit)

2 Note: If a status is specified for port operation, the high-order 8 bits of the address should be "X".

3 Note: If a new setting is made after an event clear, the event goes to the ON state.

4 Note: A new event condition has precedence over a pre-set event condition, but only at that particular setting.
EXAMPLE: 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)

DISPLAY AND EVENT STATUS;

>EV ST=OF
>EV
(ON)
STATUS = OF
ADDRESS = AXX0 (1010_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 (0000_0000_1111_1111)
DATA = 12 (0001_001)
The FILL command is used to fill memory with specified data, either hexadecimal or ASCII codes.

**Example**

FILL MEMORY WITH "0" FROM ADDRESS 0000 TO 00FF;

>F 0000,00FF,0

FILL MEMORY WITH ASCII CHARACTER FROM 0 TO FF;

F 0000,00FF,'ZAX'

1: / W SPECIFIES TO FILL MEMORY ON A WORD BASIS (DEFAULT IS A BYTE BASIS)

2 SPECIFIES THE BEGINNING ADDRESS OF THE MEMORY TO BE FILLED

3 SPECIFIES THE ENDING ADDRESS OF MEMORY TO BE FILLED (IF OMITTED, DATA TO BE FILLED FROM THE BEGINNING MEMORY ADDRESS IS WRITTEN)

4 SPECIFIES DATA TO BE FILLED WITHIN THE BEGINNING AND ENDING ADDRESSES

HEX DATA ASCII 32
EXAMPLE

Execute a user program starting at 0H;

> G 0, B

<table>
<thead>
<tr>
<th>PC</th>
<th>MC</th>
<th>OP</th>
<th>SP</th>
<th>AF</th>
<th>BC</th>
<th>DE</th>
<th>HL</th>
<th>IX</th>
<th>IY</th>
<th>I</th>
<th>IF(SP)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000B</td>
<td>1100B0</td>
<td>LD</td>
<td>DE, 0B0000H</td>
<td>FFFC</td>
<td>0000</td>
<td>0000</td>
<td>B000</td>
<td>A000</td>
<td>0000</td>
<td>00</td>
<td>0 0400</td>
</tr>
</tbody>
</table>

< Break Hardware A >

Continue the execution after break;

> G

<table>
<thead>
<tr>
<th>PC</th>
<th>MC</th>
<th>OP</th>
<th>SP</th>
<th>AF</th>
<th>BC</th>
<th>DE</th>
<th>HL</th>
<th>IX</th>
<th>IY</th>
<th>I</th>
<th>IF(SP)</th>
</tr>
</thead>
<tbody>
<tr>
<td>15DC</td>
<td>3A003A</td>
<td>LD</td>
<td>A, (3A00H)</td>
<td>FFB6</td>
<td>3A00</td>
<td>1410</td>
<td>B001</td>
<td>9F01</td>
<td>0000</td>
<td>0000</td>
<td>00</td>
</tr>
</tbody>
</table>

< Break Monitor >
**SPECIFIES THE PROGRAM EXECUTION STARTING MEMORY ADDRESS (IF OMITTED, THE PROGRAM EXECUTION IS CONTINUED FROM THE CURRENT PC)**

**SPECIFIES THE PROGRAM EXECUTION ENDING MEMORY ADDRESS**

**SPECIFIES A SECOND (OPTIONAL) ENDING MEMORY ADDRESS THAT MAY BE USED IN CONJUNCTION WITH THE FIRST ENDING MEMORY ADDRESS**

**NOTE:** WHEN **1** AND **2** ARE OMITTED, A BREAK IS NOT SPECIFIED.

**The GO command is used to execute the user program.**

**NOTE:** **G**

ALLOWS SUCCESSIVE EXECUTION OF USER PROGRAM ONCE A BREAK HAS OCCURRED
• Type 1 Real Time Trace Status
  > H cr

• Type 2 Real Time Trace Counter Reset
  > H C L R cr

• Type 3 Monitor Trigger
  > H 1 2 cr

• Type 4 Event Trigger
  > H 1 2 cr

• Type 5 Format Display
  > H 1 2 3 cr

• Type 6 History Search
  > H S 1 2 3 cr

3 ①: B M specifies the BEGIN MONITOR trigger. In this mode, trace section is initiated by the emulation start and terminated at a predetermined break point specified by the range; ②

E M specifies the END MONITOR trigger. Emulation is initiated using the GO or NEXT command to start trace. The monitor will respond with a break when the trace is terminated. The trace range up to where the breakpoint occurs is a maximum of 2048.

3 ② specifies the trace range (decimal 1 to 2047)

4 ①: B E specifies the BEGIN EVENT trigger. In this mode, trace begins when an event point is passed after emulation has begun.

C E specifies the CENTER EVENT trigger. In this mode, the trace section is recognized both before and after the event point.

M E specifies the MULTIPLE EVENT mode. In this mode, a trace is performed each time an event point is passed during loop processing.

E E specifies the END EVENT trigger. In this mode, emulation is initiated using the GO or NEXT command to start the trace. When the event point is passed, the trace terminates.

4 ② specifies the trace range (decimal 1 to 2047). When ② is omitted in the specification of BE, EM & CE, the default is 2047 and 1023 when omitted for ME.
1. M specifies the display in machine cycle format.
2. D specifies the display in machine cycle and disassemble (except for operation code fetch).
3. S specifies the display or search initiation storage pointer (decimal 1 to 2047).
4. 3 specifies the display or search termination storage pointer (decimal 1 to 2047).
5. S specifies to search for the realtime trace contents.
6. A, B, C specifies a status to be searched for:
   - A specifies the search address (PHY 16).
   - B specifies the search data (HEX DATA).
   - C specifies search status by name; M R memory read cycle, I A interrupt acknowledge cycle, M W memory write cycle, O F operation code fetch, P R port read cycle, P W port write cycle, H A halt acknowledge cycle.

The HISTORY command is used to set a trigger mode for realtime tracing, and to display or search for the user program operation to be traced. There are six different HISTORY commands available to the user: Real Time Trace Status, Real Time Trace Counter Reset, Monitor Trigger, Event Trigger, Format Display and History Search.

See APPENDIX A

A : AL B C CO D DI E EV F G H I ID L M MA N O P PI PR Q R S SA SU T U V
**EXAMPLE: 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

<table>
<thead>
<tr>
<th>POINT</th>
<th>ADDR</th>
<th>DT</th>
<th>ST</th>
</tr>
</thead>
<tbody>
<tr>
<td>0020</td>
<td>0003</td>
<td>11</td>
<td>M1</td>
</tr>
<tr>
<td>0019</td>
<td>0004</td>
<td>00</td>
<td>MR</td>
</tr>
<tr>
<td>0018</td>
<td>0005</td>
<td>04</td>
<td>MR</td>
</tr>
<tr>
<td>0017</td>
<td>0006</td>
<td>E5</td>
<td>M1</td>
</tr>
<tr>
<td>0016</td>
<td>0085</td>
<td>03</td>
<td>MN</td>
</tr>
<tr>
<td>0015</td>
<td>0084</td>
<td>00</td>
<td>MN</td>
</tr>
<tr>
<td>0014</td>
<td>0007</td>
<td>D5</td>
<td>M1</td>
</tr>
<tr>
<td>0013</td>
<td>0083</td>
<td>04</td>
<td>MN</td>
</tr>
<tr>
<td>0012</td>
<td>0082</td>
<td>00</td>
<td>MN</td>
</tr>
<tr>
<td>0011</td>
<td>0008</td>
<td>21</td>
<td>M1</td>
</tr>
<tr>
<td>0010</td>
<td>0009</td>
<td>00</td>
<td>MR</td>
</tr>
<tr>
<td>0009</td>
<td>000A</td>
<td>A0</td>
<td>MR</td>
</tr>
<tr>
<td>0008</td>
<td>000B</td>
<td>11</td>
<td>M1</td>
</tr>
<tr>
<td>0007</td>
<td>000C</td>
<td>00</td>
<td>MR</td>
</tr>
<tr>
<td>0006</td>
<td>000D</td>
<td>B0</td>
<td>MR</td>
</tr>
<tr>
<td>0005</td>
<td>000E</td>
<td>7E</td>
<td>M1</td>
</tr>
<tr>
<td>0004</td>
<td>A000</td>
<td>3A</td>
<td>MR</td>
</tr>
<tr>
<td>0003</td>
<td>000F</td>
<td>77</td>
<td>M1</td>
</tr>
<tr>
<td>0002</td>
<td>A000</td>
<td>3A</td>
<td>MN</td>
</tr>
</tbody>
</table>
EXAMPLE: HISTORY Command

DISPLAY THE REALTIME TRACE IN MACHINE CYCLE AND DISASSEMBLE CODE;

>HD

<table>
<thead>
<tr>
<th>POINT</th>
<th>ADDR</th>
<th>DT</th>
<th>ST</th>
<th>OP</th>
</tr>
</thead>
<tbody>
<tr>
<td>0023</td>
<td>0000</td>
<td>210003</td>
<td>ST</td>
<td>LD HL, 0300H</td>
</tr>
<tr>
<td>0020</td>
<td>0003</td>
<td>110004</td>
<td>ST</td>
<td>LD DE, 0400H</td>
</tr>
<tr>
<td>0017</td>
<td>0006</td>
<td>E5</td>
<td>ST</td>
<td>PUSH HL</td>
</tr>
<tr>
<td>0016</td>
<td>0085</td>
<td>03</td>
<td>MW</td>
<td></td>
</tr>
<tr>
<td>0015</td>
<td>0084</td>
<td>00</td>
<td>MW</td>
<td></td>
</tr>
<tr>
<td>0014</td>
<td>0007</td>
<td>D5</td>
<td>MW</td>
<td></td>
</tr>
<tr>
<td>0013</td>
<td>0083</td>
<td>04</td>
<td>MW</td>
<td></td>
</tr>
<tr>
<td>0012</td>
<td>0082</td>
<td>00</td>
<td>MW</td>
<td></td>
</tr>
<tr>
<td>0011</td>
<td>0008</td>
<td>2100A0</td>
<td>MW</td>
<td></td>
</tr>
<tr>
<td>0008</td>
<td>000B</td>
<td>1100B0</td>
<td>MW</td>
<td></td>
</tr>
<tr>
<td>0005</td>
<td>000E</td>
<td>7E</td>
<td>MW</td>
<td></td>
</tr>
<tr>
<td>0004</td>
<td>A000</td>
<td>3A</td>
<td>MR</td>
<td></td>
</tr>
<tr>
<td>0003</td>
<td>000F</td>
<td>77</td>
<td>MR</td>
<td></td>
</tr>
<tr>
<td>0002</td>
<td>A000</td>
<td>3A</td>
<td>MW</td>
<td></td>
</tr>
</tbody>
</table>

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

>H S, //MR

<table>
<thead>
<tr>
<th>POINT</th>
<th>ADDR</th>
<th>DT</th>
<th>ST</th>
<th>OP</th>
</tr>
</thead>
<tbody>
<tr>
<td>0013</td>
<td>0101</td>
<td>00</td>
<td>MR</td>
<td></td>
</tr>
<tr>
<td>0012</td>
<td>0102</td>
<td>00</td>
<td>MR</td>
<td></td>
</tr>
<tr>
<td>0010</td>
<td>0104</td>
<td>00</td>
<td>MR</td>
<td></td>
</tr>
<tr>
<td>0009</td>
<td>0105</td>
<td>60</td>
<td>MR</td>
<td></td>
</tr>
<tr>
<td>0007</td>
<td>A000</td>
<td>FF</td>
<td>MR</td>
<td></td>
</tr>
</tbody>
</table>

NOTE: The real time counter counts the CPU state number (CPU clock) in 32 bits.

NOTE: The pointer at which a program is initiated with a Go or Next command and the point at which the program was previously started, are displayed as the real time trace starting storage pointers. If these pointers exceed 2047, a full message is displayed.
0 specifies either the system, partial, or all incircuit mode;

0 the 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 the partial incircuit mode enables debugging using the ICD program and target system memories. The specified mapping becomes valid as do the system I/O interrupt signals.

2 the all incircuit mode enables debugging using only the target system memory. Memories specified as read/write and read only (R/W, RO) are operated by the user memory (US).

Note: When no parameter is specified, the current state is displayed.

Example

Specify the partial and system incircuit modes:

> I 1
> I
In-circuit Mode 1

> I Ø
> I
In-circuit Mode Ø

The incircuit command is used to set the ICD mapping mode.

See Appendix B
The IDENTIFICATION command is used to display an ICD device name and a version of the installed firmware.

Example

> ID
ICD-278 FOR Z80 V1.0
The Load command is used to load either a HEX file on a diskette or the object program (INTEL format) from either specified port in memory.
> M 1 9 2 9 3 9 OPT CR

1. Specifies the memory address at which the data transfer is initiated.

2. Specifies the memory address at which the data transfer is terminated.

3. Specifies the beginning memory address to which the data transfer is made.

OPT: 1 U P specifies to move data from the target memory to the ICD program memory.

P U specifies to move data from the ICD program memory to the target system memory.

Note: When OPT is omitted, data is moved according to the specification of the memory map.

The MOVE command is used to move memory between the ICD and the target system.

Example:

Move memory of 0 through FFFH;

>M 0,FFF,0,UP
### MAP Command

**MAP** is used to set the ICD memory map. The target system or ICD program memory is used on a 1K-byte basis and must be specified.

#### 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
  ```

---

| Specify the beginning address to designate mapping | PHYS 16 |
| Specifies the map ending address | PHYS 16 L BYTE |
| Specifies the program memory, either: | |
| Read only | R O |
| Read/write | R W |
| The area in which no memory exists | N O |
| The target system memory | U S |

---

1. **Note:** When 2 is omitted, 1K bytes are specified.
2. **Note:** A break occurs in this area if accessed by a program.
3. **Note:** If no parameter is specified, the current mapping condition is displayed.
EXAMPLE

Perform a single step trace, five steps from the current PC;

>N 5

<table>
<thead>
<tr>
<th>PC</th>
<th>MC</th>
<th>OP</th>
<th>SP</th>
<th>AF</th>
<th>BC</th>
<th>DE</th>
<th>HL</th>
<th>IX</th>
<th>IY</th>
<th>I</th>
<th>IF(SP)</th>
</tr>
</thead>
<tbody>
<tr>
<td>010</td>
<td>3A00A0</td>
<td>LD</td>
<td>002E</td>
<td>050E</td>
<td>A50E</td>
<td>0301</td>
<td>4312</td>
<td>0000</td>
<td>0000</td>
<td>00</td>
<td>0001A</td>
</tr>
<tr>
<td>0103</td>
<td>77</td>
<td>LD</td>
<td>(HL),A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0104</td>
<td>FE05</td>
<td>CP</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0106</td>
<td>C20001</td>
<td>JP</td>
<td>NZ,010H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0109</td>
<td>2100A0</td>
<td>LD</td>
<td>HL,0A00H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Perform a single step trace for five steps changing PC;

>R PC,200
>N 5

<table>
<thead>
<tr>
<th>PC</th>
<th>MC</th>
<th>OP</th>
<th>SP</th>
<th>AF</th>
<th>BC</th>
<th>DE</th>
<th>HL</th>
<th>IX</th>
<th>IY</th>
<th>I</th>
<th>IF(SP)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0200</td>
<td>3A003A</td>
<td>LD</td>
<td>A,(3A00H)</td>
<td>002E</td>
<td>3A42</td>
<td>A50E</td>
<td>0301</td>
<td>A000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td>0203</td>
<td>00</td>
<td>NOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0204</td>
<td>3A003A</td>
<td>LD</td>
<td>A,(3A00H)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0207</td>
<td>00</td>
<td>NOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0208</td>
<td>3A003A</td>
<td>LD</td>
<td>A,(3A00H)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**NOTE:** The starting address can be changed by changing PC with a register command.
> N 0 cr

EXAMPLE

1 SPECIFIES THE NUMBER OF SINGLE STEPS
(0 TO 15535; IF OMITTED, 1 IS ASSUMED)

N THE NEXT COMMAND IS USED TO DISPLAY ALL TRACE DATA BY N STEPS FROM THE CURRENT POINTER.
> O & 1 = 2

1. Specifies one of four offset registers, either:
   - 0, 1, 2, 3
2. Specifies the offset register set value

Example:

<table>
<thead>
<tr>
<th>O</th>
<th>80</th>
<th>81</th>
<th>82</th>
<th>83</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;O</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>A000</td>
</tr>
</tbody>
</table>

NOTE: If no parameter is specified, the current set value is displayed.

The offset command is used to set a value in an offset register for relative addressing. All of the offset registers can be used for the ICD memory addressing parameters.
The Port Command is used to change the port contents with ASCII data or hexadecimal numbers. Two types of PORT commands exist; one examines and changes the port contents and the other examines the port contents only.
The PIN command is used to mask or unmask the target system interruption signal when the IN-CIRCUIT mode is 1. When the IN-CIRCUIT mode is 0, all the interruptions of the target system are ignored and are valid when the IN-CIRCUIT mode is 2.

**EXAMPLE**

Display PIN status:

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

Disable BUSRQ and display the status:

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

**NOTE:** If no parameter is specified, the current BUSRQ, NMI and INTR conditions are displayed as masked.

**PI**

The PIN command is used to mask or unmask the target system interruption signal when the IN-CIRCUIT mode is 1. When the IN-CIRCUIT mode is 0, all the interruptions of the target system are ignored and are valid when the IN-CIRCUIT mode is 2.
The PRINT command is used to output characters sent from the ICD to the terminal and host/aux ports or the console.

EXAMPLE

Turn print on and off:

>PR ON
>
>PR OFF
The QUIT command is used to return control back to the Host Computer System. This command is valid only when the utility software program, ZICE, is available. When operational, the "04" code is transmitted to the Host Computer System.

EXAMPLE

Enable the QUIT command:

>Q
EXAMPLE

DISPLAY REGISTER WITH TITLES;

> R

PC  SP  SZHPCG A  BC  DE  HL  IX  IY  A'F'  B'C'  D'E'  H'L'  I  IF  (SP)(HL)
0000 0000 000000 00 0000 0000 0000 0000 0000 0000 0000 0000 00 0 0021 21

CHANGE HL TO A VALUE OF A000;

> R  HL=A000
> R

PC  SP  SZHPCG A  BC  DE  HL  IX  IY  A'F'  B'C'  D'E'  H'L'  I  IF  (SP)(HL)
0000 0000 000000 00 0000 0000 0000 0000 0000 0000 0000 0000 00 0 0021 3A
The REGISTER command is used to change the Z80 CPU register and program counter. There are three types of REGISTER commands available: Display, Reset and Change.

- TYPE 1 REGISTER STATUS (CURRENT REGISTER CONTENTS)
- TYPE 2 REGISTER RESET (INITIALIZES REGISTER TO 0)
- TYPE 3 REGISTER CHANGE (SPECIFIED CHANGE BELOW)

1. SPECIFIES ANY OF THE REGISTERS:
   - 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', DE', HL'
   - CY, S, Z, HC, P, N, IF(IF1)

2. SPECIFIES THE REGISTER CHANGE DATA

3. NOTE: WHEN NO PARAMETER IS SPECIFIED, THE CURRENT CONTENTS OF ALL THE REGISTERS ARE DISPLAYED.
THE SEARCH COMMAND IS USED TO SEARCH FOR THE MEMORY CONTENTS AND DISPLAY THE SPECIFIED MATCHED OR UNMATCHED ADDRESS.

EXAMPLE

>S 0,L30,12
0012

>S/W 100,110,1234

>S/D 0,10,00
0001
0002
0003

0009
000A
000B

0010
**SA**

The **SA** 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.

1. **S**pecifies which port to consider, either the:
   - **T**erminal port (ignore software handshake)
   - **P**ort (perform software handshake)
   - **A**uxiliary port (ignore software handshake)
   - **H**ost port (perform software handshake)

2. **S**pecifies the name of the file to create.

3. **S**pecifies memory address to initiate object creation.

4. **S**pecifies memory address to terminate object creation.

5. **S**pecifies the start address of the user program.

6. **S**pecifies the save message.

1. **N**ote: This parameter is valid only when the utility software program, ZICE, is available.

**EXAMPLE**

Generate the TEST.HEX file on diskette;

>SA TEST,0,37FF,0

See APPENDIX D
EXAMPLE

Check the TERMINAL port input state;

```assembly
&D1 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
```

- **NOTE:** Only a break set with a BREAK command can be used as a supervisor call.
- **NOTE:** If a supervisor call and a separate specified break exist at the same point, the supervisor call will not be executed.
> SU 1 2 cr

1. SPECIFIES EITHER:
   / C SET HARDWARE BREAK C AS A SUPERVISOR CALL
   / 7 SET SOFTWARE BREAK 7 AS A SUPERVISOR CALL
   / U SET USER SOFTWARE BREAK AS A SUPERVISOR CALL

2. SPECIFIES EITHER:
   O N SET BREAK IF USED AS A SUPERVISOR CALL
   O F F SET BREAK IF NOT USED AS A SUPERVISOR CALL

• NOTE: If no parameter is specified, the current set state is displayed.

EXAMPLE

THE SUPERVISOR COMMAND IS USED TO SET ANY GIVEN BREAK POINT AS A SUPERVISOR CALL. WHEN EXECUTED, DATA IS DIRECTLY TRANSFERRED (WITHOUT STOPPING AT THE SUPERVISOR BREAKPOINT) BETWEEN THE PROGRAM AND THE ICD SERIAL INTERFACE.

• NOTE: The function code is stored in the E register and input/output data is executed in the A register.

C See APPENDIX C
EXAMPLE

Set the trace mode;
>TA, 100,405

Display trace status;
>T (ON) ALL O100-O300

Execute trace example;
>G 100

<table>
<thead>
<tr>
<th>PC</th>
<th>MC</th>
<th>OP</th>
<th>SP</th>
<th>AF</th>
<th>BC</th>
<th>DE</th>
<th>HL</th>
<th>IX</th>
<th>IY</th>
<th>I</th>
<th>IF(SP)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0100</td>
<td>2100A0</td>
<td>LD HL, $0A0000H</td>
<td>0002</td>
<td>0023</td>
<td>1481</td>
<td>1501</td>
<td>0000</td>
<td>0000</td>
<td>00</td>
<td>0</td>
<td>8185</td>
</tr>
<tr>
<td>0103</td>
<td>1100B0</td>
<td>LD DE, $300000H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0106</td>
<td>D5</td>
<td>PUSH DE</td>
<td>0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0107</td>
<td>110003</td>
<td>LD DE, $3000H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>010A</td>
<td>7E</td>
<td>LD A, (HL)</td>
<td>FF23</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>010B</td>
<td>12</td>
<td>LD (DE), A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>010C</td>
<td>13</td>
<td>INC DE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>301</td>
</tr>
<tr>
<td>010D</td>
<td>C1</td>
<td>POP BC</td>
<td>0002</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>010E</td>
<td>C30004</td>
<td>JP $400H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0400</td>
<td>018016</td>
<td>LD BC, $1680H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0403</td>
<td>80</td>
<td>ADD A, B</td>
<td>1511</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0404</td>
<td>03</td>
<td>INC BC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1681</td>
</tr>
</tbody>
</table>

Release the trace mode;
>TC LR

- **NOTE:** Trace displays ">" with the <ESC> key to accept a command.
- **NOTE:** Trace is executed by the "GO" command, which starts the display.
SPECIFIES THE SINGLE STEP MODE (IN THIS MODE, THE ALL OR JUMP TRACE IS EXECUTED EVERY TIME THE SPACE BAR OR RETURN KEY IS PRESSED)

SPECIFIES EITHER;

A ALL MODE; ALL COMMANDS ARE TRACED AND DISPLAYED
J JUMP MODE; ONLY BRANCH COMMAND IS TRACED AND DISPLAYED

SPECIFIES THE BEGINNING ADDRESS OF THE MEMORY AREA TO BE TRACED AND DISPLAYED (DEFAULT IS 0)

SPECIFIES THE ENDING ADDRESS OF THE MEMORY AREA TO BE TRACED AND DISPLAYED (DEFAULT IS FFFF)

MAKES THE CURRENT TRACE MODE;

ON Valid
OFF Invalid
CLR Cleared

NOTE: IF NO PARAMETER IS SPECIFIED, THE CURRENT TRACE MODE SET STATE IS DISPLAYED.

THE TRACE COMMAND IS USED TO SET THE ICD TRACE MODE.
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
| A | AL | B | C | CO | D | DI | E | EV | F | G | H | I | ID | L | M | MA | N | O | P | PI | PR | Q | R | S | SA | SU | T | U | V |

> **U**

The **USER** command allows the console terminal to be used temporarily as the terminal for the **Host Computer System**. When this command is executed, the **ICD** sends all codes (excluding terminator codes received from the terminal port) to the **HOST** port without assuming them to be **ICD** commands.

1. Specifies the code used as a terminator

- **NOTE:** The following **cannot** be specified as the terminator code:
  - ESC, NAK, SP, BS, cr
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.

The VERIFY command is used to compare a host computer file or object in the Intel format, with the contents of the 10D memory.

See Appendix D
<table>
<thead>
<tr>
<th>ERROR MESSAGE</th>
<th>COMMAND OCCURRENCE</th>
<th>DISPLAYED WHEN:</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNABLE BREAK ADDRESS</td>
<td>BREAK; GO</td>
<td>WHEN A SOFTWARE BREAK IS SPECIFIED IN THE NONRAM AREA</td>
</tr>
<tr>
<td>MULTI BREAK ADDRESS</td>
<td>BREAK, GO</td>
<td>WHEN A SOFTWARE BREAK IS DUPLICATED AT THE SAME ADDRESS</td>
</tr>
<tr>
<td>WARNING UNABLE SOFT BREAK</td>
<td>BREAK</td>
<td>IF A SOFTWARE BREAK IS SET AT THE ADDRESS PRESENTLY NOT MAPPED IN RAM</td>
</tr>
<tr>
<td>*** FILE NOT FOUND</td>
<td>ALL</td>
<td>WHEN NO FILE EXISTS</td>
</tr>
<tr>
<td>*** DISK READ ERROR</td>
<td>ALL</td>
<td>WHEN DISK READ ERROR OCCURS</td>
</tr>
<tr>
<td>*** CHECK SUM ERROR</td>
<td>ALL</td>
<td>WHEN SUM CHECK ERROR OCCURS</td>
</tr>
<tr>
<td>*** DISK WRITE ERROR</td>
<td>ALL</td>
<td>WHEN DISK WRITE ERROR OCCURS</td>
</tr>
<tr>
<td>*** NO DIRECTORY SPACE</td>
<td>ALL</td>
<td>WHEN NO BLANK AREA IS AVAILABLE</td>
</tr>
<tr>
<td>ERROR MESSAGE</td>
<td>COMMAND OCCURRENCE</td>
<td>DISPLAYED WHEN:</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>--------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>C?&gt;</td>
<td>ALL</td>
<td>WHEN A COMMAND CODE ERROR OCCURS; WHEN A PARAMETER CODE ERROR OCCURS; WHEN A MODIFIER CODE ERROR OCCURS; --- WITH A &lt;BELL&gt; (CONTROL +G), WHEN THE INPUT COMMAND STATEMENT CONTAINS AN ERROR</td>
</tr>
<tr>
<td>P?&gt;</td>
<td>ALL</td>
<td></td>
</tr>
<tr>
<td>/?&gt;</td>
<td>ALL</td>
<td></td>
</tr>
<tr>
<td>MEMORY WRITE ERROR AT XXXX</td>
<td>ASSEMBLE, EXAMINE, LOAD MOVE, FILL</td>
<td>DUE TO MEMORY MODIFICATION ERROR (XXXX - IS THE MEMORY ADDRESS AT WHICH THE ERROR OCCURS)</td>
</tr>
<tr>
<td>MEMORY TIMEOUT ERROR AT XXXX</td>
<td>ASSEMBLE, DUMP, EXAMINE FILL, LOAD, MOVE, COMPARE, VERIFY, SAVE PORT, SEARCH</td>
<td>IF MEMORY I/O IN THE TARGET SYSTEM DOES NOT RESPOND TO ICD ACCESS, TIMEOUTS A WAITSTATE</td>
</tr>
<tr>
<td>I/O TIMEOUT ERROR AT XX</td>
<td></td>
<td></td>
</tr>
<tr>
<td>XXXX INPUT ERROR</td>
<td>EXAMINE, ASSEMBLE, PORT</td>
<td>AS AN INPUT ERROR, RE-ENTER DATA AFTER ERROR MESSAGE.</td>
</tr>
<tr>
<td>BREAK BUSY</td>
<td>BREAK, GO</td>
<td>WHEN THE BREAK SPECIFICATION EXCEEDS THE LIMIT</td>
</tr>
</tbody>
</table>
APPENDIX A
HISTORY Command Supplement

REAL-TIME TRACE SPECIFICATIONS

WORD WIDTH : Maximum 32 bits
WORD SIZE : 2K words
ACQUISITION CYCLE : CPU machine cycle (150 ns min.)
EFFECTIVE TRACE SECTION : 2046 machine cycles
(1.09 ms min.; 6 MHz)
FIXED TRACE DATA : A0-A15, MREQ, IORQ, RD, WR, M1, S0, S1

TRIGGERING : The start and end of tracing coordinated with the start and end of emulation can be specified, as can the event triggered delays.

DISPLAY : Machine cycle display or inverted assembly display.

REAL-TIME COUNTER SPECIFICATIONS

COUNT BIT WIDTH : 32 bits
COUNT CLOCK : CPU clock (number of states)
EFFECTIVE COUNT TIME : 716 sec.; 6 MHz

TRIGGER MODES
An event trigger with an event point specification or a monitor trigger with a breakpoint specification, is used to designate the trace start and end points. Six trigger modes are available with the ICD HISTORY command.
BEGIN MONITOR

Tracing begins at the start of emulation (initiated by a GO command) and the end of tracing is automatically assumed by the tracing range. The tracing range is specified by a delay setting of up to 2K cycles after emulation has begun.

END MONITOR

Tracing begins at the start of emulation (initiated by a GO command) and ends when control is back with the monitor on break. Note: The trace section is 2K cycles maximum before the breakpoint.
BEGIN EVENT

BEGIN EVENT
Tracing begins on passing the event point after the start of emulation (initiated by a GO command) and ends at the end of the tracing range. The tracing range is specified by a delay setting of up to 2K cycles immediately after passing the first event point.

CENTER EVENT

CENTER EVENT
Tracing begins at the start of emulation (initiated by a GO command). The trace section is specified on either side of the event point (up to 2K maximum). The end of the trace is determined by a delay of up to 2K cycles and the beginning of the trace is specified by a maximum of up to 2K cycles minus the delay specification.
**END EVENT**

END EVENT
Tracing begins at the start of emulation (initiated by a GO command) and ends after passing the first event point. The trace section is a maximum of 2K cycles up to the event point.

---

**MULTIPLE EVENTS**

MULTIPLE EVENTS
Tracing is performed each time an event is passed where multiple events are handled as in loop processing. The trace setting is specified by a delay setting of up to 2K cycles immediately after passing the event point.
## APPENDIX B

### IN-CIRCUIT Map/Pin Setting

<table>
<thead>
<tr>
<th>INCIRCUIT MODE</th>
<th>Command set</th>
<th>Mapping set Command</th>
<th>Pin set Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>I 0 System Mode</td>
<td>RO RW US NO EN DI</td>
<td>RO RW (RW) NO (DI) DI</td>
<td></td>
</tr>
<tr>
<td>I 1 Partial Incircuit Mode</td>
<td>RO RW US NO EN DI</td>
<td>(US) (US) US NO EN (EN)</td>
<td></td>
</tr>
<tr>
<td>I 2 Full Incircuit Mode</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**IN-CIRCUIT COMMAND MAP/PIN SETTING**

(RW) : OPERATES AS RW REGARDLESS OF MAP COMMAND SETTING.

(US) : OPERATES AS US REGARDLESS OF MAP COMMAND SETTING.

(EN) : OPERATES IN EN STATE REGARDLESS OF THE PIN COMMAND SETTING.

(DI) : OPERATES IN DI STATE REGARDLESS OF THE PIN COMMAND SETTING.
FUNCTION CODE KEY

(1) Function code 00 or 10: On character input from the port.

Input parameters:
Register Register E: 00H specifies the TERMINAL port.
10H specifies the HOST/AUX port.

End state:
Register Register A: Entry characters

Characters input from the port specified with an input parameter are stored in register A. If there is no character input from the specified supervisor call, control does not return to the program until character input occurs.

(2) Function code 01 or 11: Port input state signal fetch.

Input parameter:
Register Register E: 01H specifies the TERMINAL port.
11H specifies the HOST/AUX port.

End state:
Register Register A: 00H there is no input data.
FFH there is input data

This function informs the system if input data from the port specified with an input parameter.

(3) Function code 02 or 12: One character output to port.

Input parameter:
Register Register E: 02H specifies the TERMINAL port.
12H specifies the HOST/AUX port.

End state:
Register Register A: Output characters

This function outputs characters stored in register A to the port specified with an input parameter. If output of the preceding data is not completed at the supervisor call, control does not return to the target program until output is completed.

(4) Function code 03 or 13: Port output state signal fetch.

Input parameter:
Register Register E: 02H specifies the TERMINAL port.
13H specifies the HOST/AUX port.

End state:
Register Register A: 00H output has not been completed.
FFH output has been completed.

This function informs the system if data from the port specified by a parameter has been completed.
# APPENDIX C

## SUPERVISOR Function Code Setting

<table>
<thead>
<tr>
<th>FUNCTION</th>
<th>FUNCTION CODE</th>
<th>DATA OUT</th>
<th>DATA IN</th>
</tr>
</thead>
<tbody>
<tr>
<td>E - reg</td>
<td>A - reg</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TERMINAL Port data in</td>
<td>0 0</td>
<td>-</td>
<td>RECEIVE DATA</td>
</tr>
<tr>
<td>HOST/AUX Port data in</td>
<td>1 0</td>
<td>-</td>
<td>RECEIVE DATA</td>
</tr>
<tr>
<td>TERMINAL Port input status read</td>
<td>0 1</td>
<td>-</td>
<td>Input status</td>
</tr>
<tr>
<td>HOST/AUX Port input status read</td>
<td>1 1</td>
<td>-</td>
<td>Input status</td>
</tr>
<tr>
<td>TERMINAL Port data out</td>
<td>0 2</td>
<td>Output data</td>
<td>-</td>
</tr>
<tr>
<td>HOST/AUX Port data out</td>
<td>1 2</td>
<td>Output data</td>
<td>-</td>
</tr>
<tr>
<td>TERMINAL Port output status read</td>
<td>0 3</td>
<td>-</td>
<td>Output status</td>
</tr>
<tr>
<td>HOST/AUX Port output status read</td>
<td>1 3</td>
<td>-</td>
<td>Output status</td>
</tr>
</tbody>
</table>

See FUNCTION CODE KEY
APPENDIX D

OBJECT I/O Procedure

Appendix D displays the Object I/O Procedure for the commands:

<table>
<thead>
<tr>
<th>LOAD</th>
<th>SAVE</th>
<th>VERIFY</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>MODE</th>
<th>REMOTE</th>
<th>LOCAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>/T</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>/P</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>/A</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>/H</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Default</td>
<td>0 (HOST)</td>
<td>X (TERMINAL)</td>
</tr>
</tbody>
</table>

0: Denotes software handshake is performed. For the correct communication protocol, please refer to:

Software Specifications of Host Computer System

Note: For the load message to be used, /T and /A must be specified at the object input port.

Note: The object name may be specified as the user definition load message if /P and /H are designated during software handshake.