NEW MICE-II USER'S MANUAL

FOR 8-BIT ZILOG

SERIES MICROPROCESSORS

- Z80, NSC800, Z8, ZS8 -

USER'S MANUAL FOR NEW MICE-II

Z80
NSC800
Z8
ZS8

Doc No. I49-000115
2nd Edition
January 1989

DISCLAIMER

This manual is subject to change without notice. Nothing herein shall be construed as a recommendation to use any product in violation of existing patents or other rights of third parties.

NEW MICE-II AND THIS MANUAL ARE COVERED BY THE LIMITED WARRANTY SUPPLIED WITH NEW MICE-II AND REPRODUCED IN APPENDIX N OF THIS MANUAL.

Printed in Taiwan, ROC
January 1989

MICROTEK INTERNATIONAL INC.
6 Industrial Road East 3
Science-Based Industrial Park
Hsinchu, Taiwan, 30077, R.O.C.
TEL: (035)772155
TLX: 32169 MICROTEK
FAX: (035)772598

PREFACE

Microtek International Inc. proudly presents the award winning Micro-In-Circuit-Emulator, NEW MICE-II. You have made an excellent purchase and will soon benefit from the many advance features that Microtek designs into all of its products.

With the detailed instructions provided in this manual, you can rapidly set up and operate your new NEW MICE-II. The information provided in this user's manual is believed accurate and complete, but no responsibility is assumed for any error or omission.

Microtek is dedicated in providing leadership in the microcomputer-based industry. We continue to design new emulators with enhanced features, low cost and always geared toward the microprocessor designer's need. Thank you for choosing Microtek; and may we suggest that you also investigate the many other fine products that Microtek has to offer.

The sale of any Microtek product is subject to all Microtek Terms and Conditions of Sale and Sales Policies, copies of which are available upon request.

Microtek supplies a variety of hi-tech equipment to the microprocessor-based industry. For further information on other Microtek products and/or any other questions regarding this manual, please contact Microtek headquarters.

NEW MICE-II

Section

TABLE OF CONTENTS

APPENDICES

CHAPTER 1

GENERAL INTRODUCTION

1.1 Introduction To MICE

Microtek's versatile Micro-In-Circuit-Emulator (MICE) is a low-cost development tool that emulates most industry-standard microprocessors. It has set new standards for universal, high-performance emulation, at an exceptionally low cost-per-function.

Traditionally, microprocessor development was done on dedicated systems. The computer system had its complement of peripheral devices and sufficient mass storage to accommodate user programs, data, etc. A dedicated emulator was attached to the system to assist the designer in ensuring that both hardware and software were functioning properly.

Recently, a variety of general purpose computers have been designed for use in microprocessor development. Through the use of cross assemblers, code can be generated for the target system. However, these emulators are still dedicated to the particular computer.

With MICE, a third and more practical approach to microprocessor development is now available; one that uses fewer resources, performs most of the same functions, yet costs only a fraction compared with its predecessors. The MICE module is controlled via an RS-232C compatible interface. All software required to operate the MICE module are contained in EPROMs within the module. MICE can be operated by using a display

terminal only or in conjunction with a computer system. Different processors can be emulated by merely changing the personality card and associated EPROMs.

Some key features of MICE are:

* Operation at speeds up to the maximum rated frequency of the specified microprocessor.

* Target processor retains its entire memory and I/O space.

* Enabling and disabling of hardware control signals to the processor with console commands.

* Resident assembler and two-pass disassembler which assigns labels to subroutine and branch addresses.

* Built-in memory diagnostics and block memory transfer for target processor memory.

* Downloading and uploading of target program between MICE and host computer system.

* Help command that lists all commands along with the proper syntax.

1.2 Introduction to NEW MICE-II

1.2.1 Enhanced Features

NEW MICE-II is an enhanced member of the MICE family. Aside from retaining the original MICE features, NEW MICE-II has substantially been upgraded to include the following new features:

* High Performance Universal Emulation Memory (HUEM) with memory size selectable at 64K or 128K bytes (2 independent 64K settings) of static RAM for 8-bit NEW MICE-II models (user must specify if required for microcontroller versions).

* Super Universal Emulation Memory (SUEM) with memory size 256K bytes of static RAM for 8 bit NEW MICE-II models.

* 8/256 Kbytes block enable/disable capability for HUEM/SUEM.

* User-qualified trigger to specify start or end of tracing for source code and machine statuses. Up to 2048 cycles and 40 channels of signals may be recorded for program debug.

* Supports up to 8 hardware trace points for program debug.

* Real-time forward and backward trace for up to 2048 machine cycles.

* Two real-time breakpoints.

* Supports disassembly for both Instruction Step and List Trace commands.

* With built-in power supply and cooling fan.

1.2.2 Breakpoint Processor

NEW MICE-II may also be equipped with the new Breakpoint Processor (BPP) board at user's option. The board allows the NEW MICE-II to provide more advanced features as summarized below:

* Sophisticated breakpoint logic, with up to 120 new breakpoint constructs.

* Flexible trigger constructs that can define single events, multiple activities or external hardware signals.

* Two data breakpoints and two external hardware breakpoints.

* Breakpoint interval timer that displays the interval between initial trigger and emulation stops.

1.2.3 Major Hardware Overview

NEW MICE-II is a low-cost powerful emulation instrument basically equipped with the following boards:

Control Emulation Processor (CEP)
- also called personality board

Real-time Trace (RTT)

High Performance Universal Emulation Memory (HUEM) or **Super Universal Emulation Memory (SUEM)**
- also called memory board

BreakPoint Processor (BPP)

Remember that the CEP board can be interchanged to form any other NEW MICE-II. If more emulation memory is needed, additional HUEM boards may be added. And where more powerful emulation breakpoint or triggering capability is required, the BPP board is available.

1.2.4 Conclusion

At half the cost of competitive units, NEW MICE-II provides features not available with other emulators. Some of the advantages of NEW MICE-II over other emulators are:

universality A wide variety of microprocessors can be emulated using this single development tool, thus avoiding both the inflexibility of dedicated systems and the heavy capital investment required for general purpose systems.

versatility Emulation of a different processor only requires changing the personality board. Multiple design projects can be carried out concurrently, and the CEP board can be reused for different projects.

flexibility With NEW MICE-II providing its own resident assembler and disassembler, it can be minimally configured with just a display terminal and power supply. After the target program has been downloaded from the host computer, testing can be done off-line, thereby freeing the computer.

1.3 NEW MICE-II Operating Configurations

NEW MICE-II can be used in several configurations. The simplest configuration requires only an RS-232C compatible terminal as the display console and command entry device. A computer system can also be configured as the controlling device.

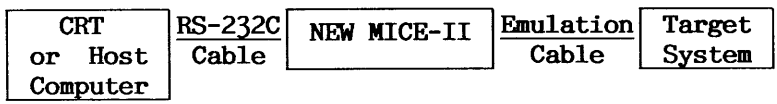


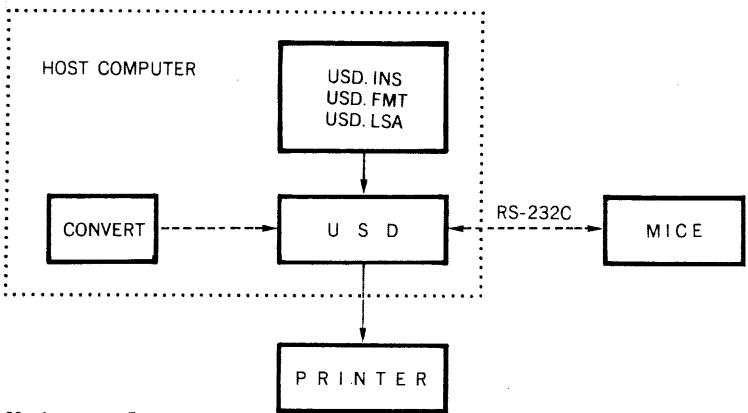
Figure 1-1
Configuring NEW MICE-II With an RS-232C
Compatible Terminal or Host Computer

When a computer system is interfaced with NEW MICE-II, a driver program must be resident in the system. Driver programs and symbolic translator drivers are available for various systems. (See Appendix E for a detailed listing.)

Microtek has access to various sources of driver programs and other software tools. For updated information contact your local Microtek representative.

1.3.1 Universal Symbolic Debugger (USD)

The USD (Universal Symbolic Debugger), is a sophisticated software package designed specifically to enhance the debugging and other capabilities of Microtek's MICE series of emulators. USD acts an interface between the MICE user, MICE, and the host computer system, to make the optimum use of the resources available on each. It enhances the range of development, emulation, analysis and debugging features of MICE to a new dimension and providing MICE users with a high quality and broad range of support to suit their various engineering applications.



* Universal
* Enhanced symbol processing capability
* File management
* Command file
* Logic state analysis
* Software performance analysis
* Sophisticated utility commands
* Fully supports object files, and MICE's

* **Universal**

Almost all popular cross-assembler/cross-compiler formats, such as 2500AD, IAR, intel OMF51/85/86, and MRI format symbol tables are accepted. MICE command for all type MICE are fully transparent.

* **Enhanced symbolic processing capability**

Symbol table file can be: loaded/listed/saved; symbols can also be created/deleted. Any address can be qualified to check for symbol equivalence.

* **File management**

Hex file can be down loaded to emulation/user memory, and data in memory can be uploaded to host computer. All MICE and USD commands can be logged as a macro command file; all debug commands as well as the command output can be logged as an output file. The logged output file is especially useful in detailed checking of sophisticated debug session.

* **Command file**

Unlimited command lines can be edited in a command file. 10 replaceable parameters are permitted; conditional subcommands and loop block capability are provided.

* **Logic state analysis**

Trace buffer data, such as address bus, data bus, control, and spare signals can all be displayed in waveform.

*** Software performance analysis**

Module entry histogram:

Display the number of calls for specified program addresses, along with a bar graph showing the percentage of processor activity for each address.

Execution interval histogram:

Display the number of times a program routine executes within different timing frames, along with a bar graph showing the percentage of processor activity for each range.

*** Sophisticated utility commands**

Help for USD commands
Window facility is provided: foreground and background windows
Up to 16 command lines can be retained
DOS commands access

*** Fully supports object files, and MICE's**

Firmware version supported (for the firmware and higher version).

For NEW MICE-II

Z80	V3.1 & later
NSC800	V3.2 & later
Z8	V3.1 & later
ZS8	V3.0 & later

*** Host computers supported**

IBM PC/XT/AT (MS-DOS 2.0)
NEC 9801 (MS-DOS 3.0)
SUN MICRO WORKSTATION (UNIX 2.0)
VAX/VMS 4.1
uVAX/ULTRIX

1.4 NEW MICE-II Applications

Because of its unique design, NEW MICE-II offers new applications which include:

1. Inexpensive evaluation of new microprocessors without purchasing a special evaluation board or expensive development system.

2. Several designers can share the use of a single development system, eliminating the difficult problem of allocating a single resource and the need for multiple work-stations. Large programs can be edited, assembled or compiled, and then downloaded to the target system. Since NEW MICE-II has its own assembler and disassembler, the programs can then be tested using only a display terminal.

3. NEW MICE-II's compact size, light weight, and rugged construction makes it an ideal field service instrument. Easily transported and set up in remote locations, NEW MICE-II can reduce downtime, provide on the spot diagnosis and resolution of field problems, avoiding customer inconvenience and expensive service delays. With its RS-232C interface, NEW MICE-II can be quickly interfaced with any compatible display terminal. Diagnostic programs can be generated using the resident assembler.

4. Personal computers can be upgraded to development systems at a fraction of the typical costs. Driver programs for various computers have already been written, allowing programs assembled or compiled to be downloaded.

1.5 NEW MICE-II Definitions

MICE is a patented trade name for Microtek emulators and stands for Micro-In-Circuit-Emulator.

ICE cable is an In-Circuit-Emulator cable that joins NEW MICE-II to the target.

NEW MICE-IIH is a conventional MICE emulator module consisting of three interconnected printed circuit boards. These boards are:

1. Control Emulation Processor (CEP) board.
2. Real-time Trace (RTT) board.
3. High Performance Emulation Memory (HUEM) board.

NEW MICE-IIS is a conventional MICE emulator module consisting of four interconnected printed circuit boards. These boards are:

1. Control Emulation Processor (CEP) board.
2. Real-time Trace (RTT) board.
3. Super Universal Emulation Memory (SUEM) board.
4. Breakpoint Processor (BPP) Board.

CHAPTER 2

NEW MICE-II INSTALLATION PROCEDURES

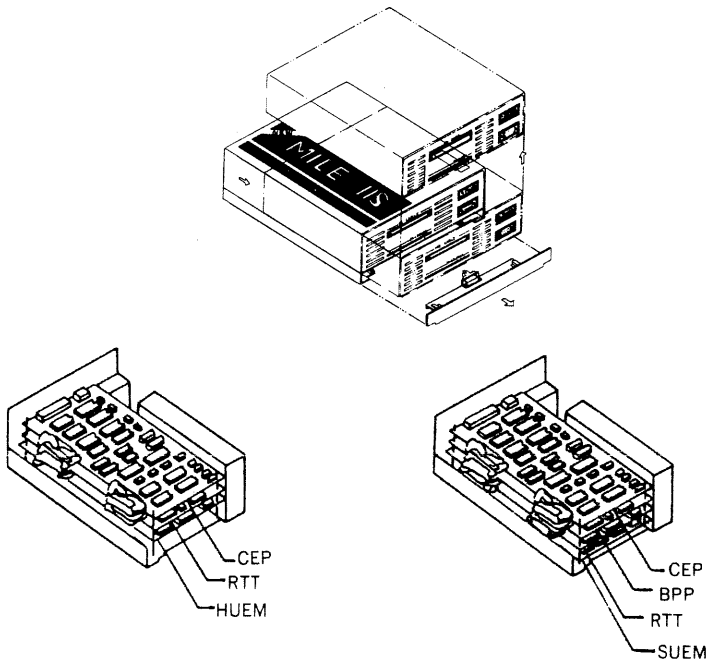
NEW MICE-II comes from the factory preset and completely assembled. The NEW MICE-II module consists of three printed circuit boards: Control Emulation Processor board (CEP), Real-time Trace board (RTT) and High Performance Universal Emulation Memory board (HUEM). ICE cables are also included which must be connected between NEW MICE-II and the target system.

2.1 Setting Up NEW MICE-II

Detailed instructions for setting up the NEW MICE-II module are as follows:

2.1.1 Opening the NEW MICE-II Case

NEW MICE-II is contained in a two piece high impact metal case. To open the case, hold the lower portion of the vertical end (rear) firmly with both hands and with your thumbs slide the top half forward about one half inch. This will separate the top from the bottom half.



NEW MICE-IIH NEW MICE-IIS

Figure 2-1
Opening The NEW MICE-II Case

2.1.2 Replacing the Personality Board (CEP) or Emulation CPU

1. To install a personality board in place of the current card, the cover of the NEW MICE-II case must first be removed. With power off, carefully remove the top personality card (CEP) from the NEW MICE-II module by disconnecting the two ribbon cable connectors which join the CEP board to the RTT board below, and then removing the six locking screws and six brass spacers.

   Install the new personality board into the module by refastening the brass spacers and locking screws to hold the replacement card permanently in position. Then reconnect the two ribbon cable connectors from the RTT board below to the CEP board; the arrows indicating pin 1 must be aligned.

2. Only one CPU is supplied with NEW MICE-II per the customer's specification. To emulate the other processor, replace the CPU as indicated below:

NEW MICE-II	CPU	Location
* Z80	Z80/Z80A/Z80B/Z80H	U36
NSC800	NSC800	U24
Z8	Z8612	U38
ZS8	Z8883	U42

   * CPU alteration applicable to Z80 only.

2.1.3 Internal Adjustments

Access to all adjustment switches is possible with the NEW MICE-II cover removed. The DIP switch on the personality board is readily accessed from the top; and the three DIP switches for memory board selection are located in the opening at the end of the NEW MICE-II case, which results when the cover is removed. Preset instructions are explained in the following sections of this chapter.

2.1.4 Connecting the ICE Cable

1. Z80

The ICE (In-Circuit-Emulator) cable assembly consists of one flat-wire cable with a 40-pin connector at one end and an IC header at the other end. To install the ICE cable after setup has been completed, remove the cover to NEW MICE-II, and attach the 40-pin connector at position J5 on the CEP board. Thread the IC header end through the rectangular opening in the NEW MICE-II case and slide the cover shut (section 2.1.6). Then attach the end with the IC header to the 40-pin target processor socket on the target board. Note that the ICE cable cannot be interchanged with other NEW MICE-II models.

2. NSC800

The ICE cable assembly (Figure 2-2) consists of an IC header at the Target Connected Board (TCB-NSC800DP). To install the ICE cable after setup has been completed, remove the cover of the NEW MICE-II, thread the connector end through the rectangular opening in the NEW MICE-II case, attach the 40-pin connetor on the CEP board and slide the cover shut. At the target end of the cable, remove the protective padding from the bottom of the IC header. Gently insert the header into the appropriate CPU socket on the target board.

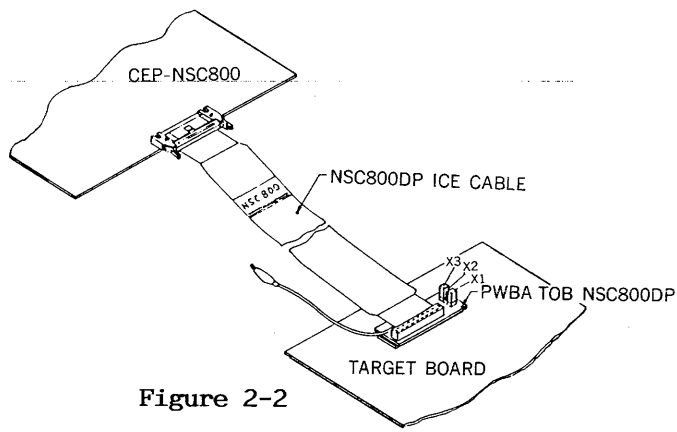


Figure 2-2

3. Z8

The ICE cable assembly (Figure 2-3) consists of two flat-wire cables with dual 40-pin connectors at one end (CHZ8) and a single 40-pin connector at the other end (THZ8). To install the ICE cable after setup has been completed, remove the NEW MICE-II cover and attach the end with dual 40-pin connectors at position J5 on the CEP board. Thread the IC header end through the

rectangular opening in the NEW MICE-II case and slide the cover shut (section 2.1.6). Then attach the end with a single 40-pin connector to the 40-pin target processor socket on the target board. (Note that the ICE cable cannot be interchanged with other NEW MICE-II models.)

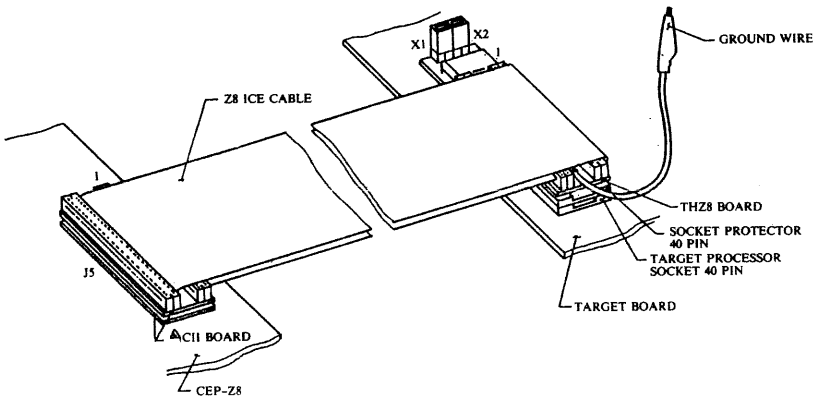


Figure 2-3

4. ZS8

Two different ICE cables are used for 40 and 48 pin target CPUs. The ICE cable assembly for a 40-pin target CPU (Figure 2-4) consists of two flat-wire cables with dual 40-pin connectors at one end and a single 40-pin connector at the other end (THZS8-40). The ICE cable assembly for a 48-pin target CPU (Figure 2-5) consists of two flat-wire cables with dual 50-pin connectors at one end and a single 48-pin connector at the other end (THZS8-48).

To install the ICE cable after setup has been com-

pleted, remove the NEW MICE-II cover and attach the end with dual 40/50 pin connectors at position J6/J5 on the CEP board. Thread the IC header end through the rectangular opening in the NEW MICE-II case and slide the cover shut (section 2.1.6). Then attach the end with a single 40/48 pin connector to the 40/48 pin target processor socket on the target board. Note that the ICE cable cannot be interchanged with other NEW MICE-II models.

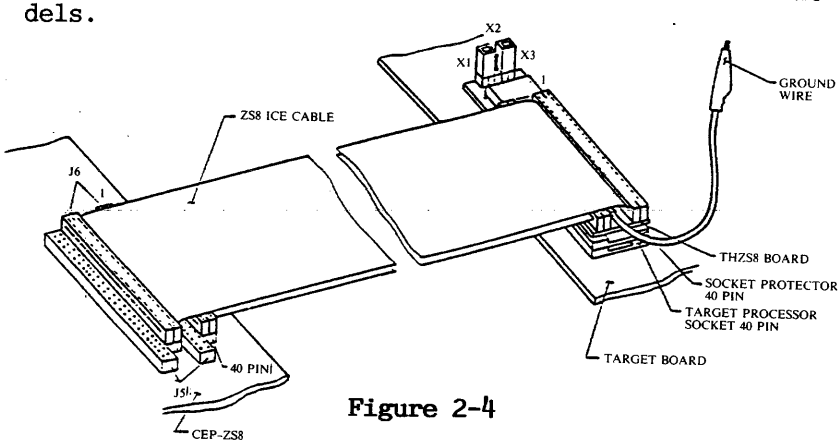


Figure 2-4

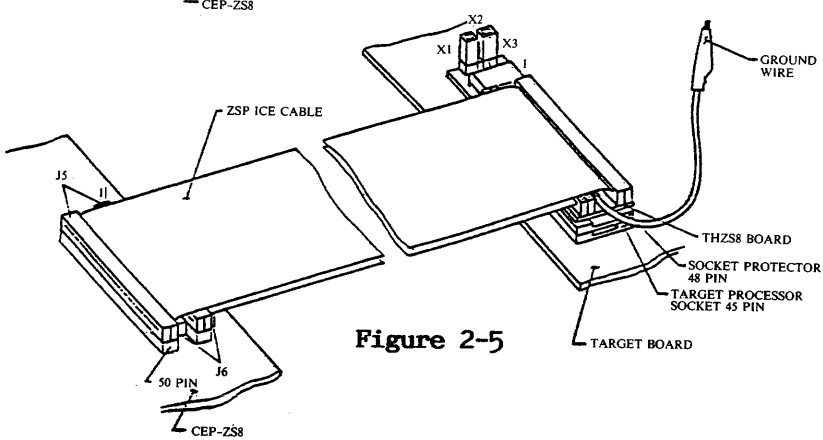


Figure 2-5

2.1.5 Connecting the External Trace Cable

The external trace cable shown below is used for recording external signals during trace operations. To monitor any external activity not automatically recorded by trace commands, the input cable must be connected from the trace point connector on the CEP board to the target. This connector consists of a 9 post stick header on the CEP board designated X0-X7 from right to left, plus XG for ground.

The monitored signals are called spare bits and support concurrent trace with the address, data and status bus. Any of these bits can be monitored by trace commands to display hardware status (in the SPARE column) by executing List, Cycle Step and Instruction Step commands; where "1" represents a high (or floating) signal and "0" indicates a low signal.

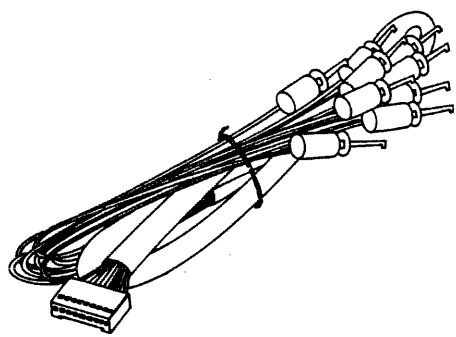


Figure 2-6
Signal Monitor Cable

2.1.6 Closing the NEW MICE-II Case

To close the cover, set it over the base leaving a 1/2 inch gap. Holding the base firmly in the front, use your thumbs to push the cover the remaining distance. This will force the locking tabs into position and firmly attach the top to the base.

2.2 NEW MICE-II Specifications

Mechanical specifications for NEW MICE-II are:

Width	26.0 cm	(10.24 in.)
Height	12.7 cm	(5.00 in.)
Length	33.0 cm	(13.00 in.)
Weight	5.0 kg	(11.00 lb.)

The minimum and maximum operation and storage limits for temperature and humidity are:

Operating Temperature: 0°~50°C(32°~122°F)
Storage Temperature: -10°~65°C(14°~149°F)
Relative Humidity: 20 ~80%

NEW MICE-II equips one built-in power supply.

Power Consumption: AC100-120V~: 1.3A MAX
AC200-240V~: 0.65A MAX
47-63HZ

Voltage requirements: AC100-120V~ or
AC200-240V~
Factory-set

2.3 Communicating with NEW MICE-II

Whether NEW MICE-II is connected to a terminal or to a computer system, the connection between the controlling device and NEW MICE-II is across a programmable RS-232C compatible interface.

2.4 Control Emulation Processor Board (CEP) Setup

Other NEW MICE-II emulators can be formed simply by replacing the personality (CEP) board. If a change of personality boards is required, or modification to the factory preset conditions is necessary, the adjustment options are described in the following sections.

2.4.1 Interface Parameter Selection

On the CEP board locate the six position DIP switch (NSC800 - U35 and all others - U17). This switch sets the following communication modes which are only examined by the control processor during power-up and reset.

Switch Selection	Description
S1-S2-S3	Baud Rate
S4	7/8 Data Bits
S5	Disable/Enable Parity
S6	Odd/Even Parity

The number of stop bits is permanently set at two; and communication is full duplex. Each data frame consists of 1 start bit, 2 stop bits, 7/8 data bits and 1 parity bit if parity is enabled.

The transmission rate can be specified from 150-19200 baud by setting S1-S2-S3 of the DIP switch as follows:

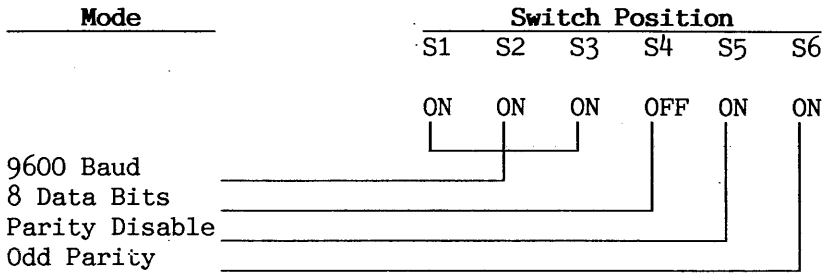
Baud Rate	Switch Section S1	S2	S3
150	ON	OFF	OFF
300	OFF	ON	OFF
600	ON	ON	OFF
1200	OFF	OFF	ON
2400	ON	OFF	ON
4800	OFF	ON	ON
9600	ON	ON	ON
19200	OFF	OFF	OFF *

* This setting is only effective for the following (or later) firmware versions: V3.0 - ZS8, V3.1 - Z80/Z8 and V3.2 - NSC800. For all previous versions, this switch setting indicates 110 baud.

The number of data bits and parity can be specified by setting S4, S5 and S6 of the DIP switch as follows:

Mode	Switch Section S4	S5	S6
8 Data Bits	OFF		
7 Data Bits	ON		
Parity Enable		OFF	
Parity Disable		ON	
Even Parity			OFF
Odd Parity			ON

NEW MICE-II personality boards are shipped from the factory preset to 9600 baud, 8 data bits, and none parity. Switch position for this default selection is as follows:



Default setting

When parity is disabled, the odd/even parity switch (S6) section can be set to either position since it is ignored.

2.4.2 Clock Selection

1. For NEW MICE-II Z80

NEW MICE-II Z80 supports the Z80 dynamic RAM refresh function and is available in the following versions:

Z80 - 2.5MHz clock oscillator
Z80A - 4MHz clock oscillator
Z80B - 6MHz clock oscillator
Z80H - 8MHz clock oscillator

The target's clock, the on-board clock, or the on-board clock with an option to support the target can be selected by placing jumpers on the personality board as follows:

	X8	X9	X10	X11
EXT	C	O	O	C
INT	O	C	C	O
INT --> TARGET	C	O	C	O

C: Close
O: Open

Internal/External Clock Selection

2. For NEW MICE-II NSC800

The target's clock or the on-board 5 MHz clock can be selected by placing jumpers on the personality board as follows:

TCB-NSC800DP Adapter

	X8	X9	X10	X11	X1	X2	X3
INT	O	O	C	O	O	O	O
EXT TTL CLOCK	C	C	O	O	O	C	O
EXT CRYSTAL	C	C	O	O	C	O	C

Internal/External Clock Selection

C: Close
O: Open

3. For NEW MICE-II Z8

The target's clock or the on-board 12MHz clock can be selected by placing jumpers on the personality board as follows:

THZ8*
Adapter

	X8	X9	X1	X2	X3
INT	C	O	O	O	O
EXT TTL CLOCK	O	C	O	C	O
EXT CRYSTAL	O	C	C	O	C

C: Close
O: Open

Internal/External Clock Selection

*The THZ8 Adapter is located on the target side of the ICE cable.

4. Clock Selection for NEW MICE-II ZS8

The target's clock or the on-board 20MHz clock can be selected by placing jumpers on the personality board as follows:

THZS8*
Adapter

	X8	X9	X1	X2	X3
INT	C	O	O	O	O
EXT TTL CLOCK	O	C	O	C	O
EXT CRYSTAL	O	C	C	O	C

C: Close
O: Open

Internal/External Clock Selection

* The THZS8 Adapter is located on the target side of the ICE cable.

2.4.3 Ready Signal Selection (Z80 and NSC800)

1. For NEW MICE-II Z80

The target ready signal, or on-board ready signal (when memory is located in SUEM/HUEM) can be selected by placing jumpers on the personality board as follows:

	X14	X15
TARGET	O	C
ON-BOARD	C	O

C: Close
O: Open

2. For NEW MICE-II NSC800

The target ready signal can be selected by removing the jumper at X14; or if MICE emulation memory is selected, the on-board ready signal can be used by connecting a jumper at X14.

2.4.4 Bus State Selection (Z80)

NEW MICE-II Z80 permits the CPU to be stopped at either WAIT or BUSAK state. Stopped at WAIT state, the two refresh signals RFSH, MREQ will be lost but the CPU will stop in the middle of the machine cycle. Therefore, the messages displayed have not yet been executed. Under this condition, the DATA BUS, ADDRESS BUS and STATUS signals are still active and can be read. This simplifies hardware debug. Stopped at BUSAK state, refresh is still supported to the target; and the CPU will stop at the end of the machine cycle where the messages displayed have already been executed. Selection is shown below:

	X12	X13	
$\overline{\text{WAIT}}$	C	O	C: Close
$\overline{\text{BUSAK}}$	O	C	O: Open

$\overline{\text{WAIT}}$/$\overline{\text{BUSAK}}$ Mode Selection

2.4.5 Vcc Selection (NSC800)

The NSC800 Vcc on the target can be selected by removing the jumper at X12; or the on-board Vcc can be used by placing a jumper at X13.

	X12	X13	
INT	O	C	C: Close
EXT	C	O	O: Open

Internal/External Vcc Selection

2.4.6 ISR Function (Z80)

CEP-Z80 revision M supports ISR (Interrupt Service Routine) function in emulation memory. Mini-jumper X24 is used to enable/disable ISR.

X24	ON	ISR disable (default)
	OFF	ISR enable

Application Notes

When applying ISR function, if RETI instruction is placed in MICE emulation memory, take limitations below:

1. In the target, data buffer is not allowed between CPU and I/O devices.

2. Target memory location must not overlap with MICE emulation memory.

3. Before setting X24 to OFF, be sure to connect X19.

It is not able to implement ISR function by reworking previous version CEP L1 or earlier versions. ISR function only applies to firmware V2.1 and later.

2.5 High Performance Universal Emulation Memory Board (HUEM) Setup

There are four DIP switches located at the front of the HUEM board (bottom board in the NEW MICE-II module); U14/U28 (enabled for both the standard 64K byte version and the optional 128K byte version) and U36/U43 (enabled only for the optional 128K byte version). The individual keys on the switches are designated S1 through S8 or S10. (Refer to the placement chart on Appendix F for the board location of these switches.)

There are two separate 64K memory banks in the HUEM. U14 and U28 set bank 1; and U36 and U43 set bank 2. Note that bank 2 is only populated for the optional 128K byte version.

The DIP switches setup are explained in the following paragraphs.

2.5.1 U14 and U36: Emulation Memory Enable/Disable

The user can select any memory block of 8K bytes to disable or enable depending on the amount of emulation memory needed. Set individual keys ON for enable and OFF for disable.

S1	S2	S3	S4	S5	S6	S7	S8
Lowest 8K ..... **Highest 8K**							

Position S9 and S10 are used as follows:

Mode	S9	S10
Write Enable	ON	
Write Protect	OFF	
Memory Enable		OFF
Memory Disable		ON

2.5.2 U28 and U43: Memory Segment Select

Memory segments may be selected within 16M bytes by setting U28 and U43 as follows:

Segment	Start Address	S1	S2	S3	S4	S5	S6	S7	S8
0K	0H	ON	ON	ON	ON	ON	ON	ON	ON
64K	10000H	OFF	ON	ON	ON	ON	ON	ON	ON
128K	20000H	ON	OFF	ON	ON	ON	ON	ON	ON
"	"				"				
"	"				"				
16320K	FF0000H	All OFF							

Note: If bank 1 and bank 2 have the same start address (i.e. U28 and U43 have the same key setting), HUEM will select bank 1 only and disregard the U43 key setting for bank 2.

2.5.3 Factory Preset

HUEM is preset at the factory with the following switch positions (0-128K bytes):

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
U14,U36	ON	ON	ON	ON	ON	ON	ON	ON	ON	OFF

In this configuration all blocks (64K byte x 2) of emulation memory are enabled and write enabled.

	S1	S2	S3	S4	S5	S6	S7	S8
U28	ON	ON	ON	ON	ON	ON	ON	ON

All U28 keys are in the ON position. In this configuration, the bank 1 memory segment start address is 0000H.

	S1	S2	S3	S4	S5	S6	S7	S8
U43	OFF	ON	ON	ON	ON	ON	ON	ON

Only S1 is OFF. In this configuration, the bank 2 memory segment start address is 10000H.

Setting U14, U28, U36 and U43 completes setup of the HUEM board.

2.6 Memory Selection for Microcontrollers (Z8 and ZS8)

2.6.1 Memory Selection for NEW MICE-II Z8

1. The Z8 family of processors has three types of memory: internal program memory (P), external program memory (P), and external data memory (X). P and X serve as optional qualifiers for M/T commands, with program memory (P) as the default value. Note that if qualifier X is used to address internal program memory, an error message will display:

"LOCATION ERROR!"

CPU	Addressable Memory			Addressable I/O
	Internal Program	External Program	External Data	
Z8611/Z8613	000H-FFFH	1000H-FFFFH	1000H-FFFFH	Ports P0-P3
Z8601/Z8603/Z8671	000H-7FFH	800H-FFFFH	800H-FFFFH	Ports P0-P3
Z8681	none	000H-FFFFH	000H-FFFFH	Ports P0-P3
Z8682*	none	800H-FFFFH	800H-FFFFH	Ports P0-P3

*000H-7FFH is not addressable.

2. Internal program memory (4K bytes) is located in emulation memory on the CEP board (6264 RAM at U51).

3. If program memory is larger than 4K bytes or external data memory is used, then a SUEM/HUEM board must be used for emulation memory. To select SUEM/HUEM, set control register P01M (F8H) with the R command, or with the user's program, to define Port 0 in address mode (P0=ADDR) and Port 1 in address/data mode (P1=A/D); i.e. set D7; D4 & D1=1 and D3=0 in register P01M (RF8=1XX10X1X).
   Example: Set RF8=96H to specify -

   a) P00-P07=A8-A15

b) P10-P17=AD0-AD7
c) internal stack pointer
d) normal external memory timing

4. However, if memory space is segregated, then input the EX command (Chapter 6), and logically select program memory and data memory using Data Memory select output. Define Port3-pin4 for Data Memory output, by setting control register P3M (F7H) with the R command, or with the user's program, to P34=DM. External segregated emulation memory is located in SUEM/HUEM, with program memory at 1000H-FFFFH and data memory at 11000H-1FFFFH.

Example: If the program memory range is 0_3FFFH and data memory range is 1000H_1FFFH, then set emulation memory as indicated below for SUEM or HUEM.

SUEM Set these two parameters regardless of the option selected -
* emulation memory block enable/disable
* memory offset address to 0H

a) Emulate program memory only by setting -
* memory segment to 0H
* starting address to 0H

b) Or emulate data memory only by setting -
* memory segment to 10000H

HUEM Enable all emulation memory by setting -
* U14 and U36 S1-S9 ON and S10 OFF
Emulate program memory in bank 1 by setting -
* memory segment to 0H (U28 S1-8 all ON)
Emulate data memory in bank 2 by setting -
* memory segment to 10000H (U43 S1 OFF, all others ON)

5. HUEM can cover the entire emulation memory range required for external program and external data memory (max 120K bytes).

Because the offset address **cannot** be defined for HUEM, the first 4K bytes of both memory banks are disabled. For information on HUEM switch settings refer to section 2.5.

Emulation Memory Selection

Required Memory Size / Emulation Memory Location	≤4K bytes	>4K bytes	
		Segregated (set EX, & P34=$\overline{DM}$)	Combined (set DX)
SUEM	Not needed	Use for program or data memory (max 120K bytes)	Max 60K bytes
HUEM	Not needed	Use for program or data memory, or both (max 120K bytes)	Max 60K bytes
CEP U51(6264)*	Used as internal program memory (Max 4K bytes)		

*If program memory (0-4K) is located in target ROM, either read the data into the host computer and then download it to MICE emulation memory, or place the target ROM (if a 2732 or 2764) on the CEP-Z8 at U51.

a) If HUEM is selected and memory is combined, then both program and data memory must be located in the same memory bank (HUEM has two separate 64K byte memory banks) with the memory segment set to OH (U37/U43 S1-8 all ON). Because the first 4K bytes (in each memory bank) are always disabled (OK-4K internal program memory is located on the CEP-Z8), the maximum addressable memory on HUEM is 60K bytes.

   However, if memory is segregated, then program memory should be located in one memory bank and data memory in the other bank. The maximum addressable memory on HUEM is then 120K bytes.

b) When using SUEM, up to 120K bytes of external memory can be defined. The start address for 120K bytes of emulation memory can be specified anywhere within the available emulation memory (refer to SUEM section).

c) The first 4K bytes are always internal program memory. Remember that the 2K to 4K bytes of external program memory downloaded to the CEP-Z8 for the Z8601, Z8603, Z8671 and Z8682 is addressed as internal memory.

2.6.2 Memory Selection for NEW MICE-II ZS8

1. The Super-8 family of processors has three types of memory: internal program memory (P), external program memory (P), and external data memory (X). P and X serve as optional qualifiers for M/T commands, with program memory (P) as the default value.

CPU	Addressable Memory			Addressable I/O
	Internal Program	External Program	External Data	
Z8811/Z8813	0000H-0FFFH	1000H-FFFFH	0000H-FFFFH	Ports P0-P3
Z8821/Z8823	0000H-1FFFH	2000H-FFFFH	0000H-FFFFH	Ports P0-P3
Z8831/Z8833	0000H-3FFFH	4000H-FFFFH	0000H-FFFFH	Ports P0-P3
Z8810/Z8812	0000H-0FFFH	1000H-FFFFH	0000H-FFFFH	Ports P0-P4
Z8820/Z8822	0000H-1FFFH	2000H-FFFFH	0000H-FFFFH	Ports P0-P4
Z8830/Z8832	0000H-3FFFH	4000H-FFFFH	0000H-FFFFH	Ports P0-P4
Z8801	none	0000H-FFFFH	0000H-FFFFH	Ports P2-P3(P0)
Z8800	none	0000H-FFFFH	0000H-FFFFH	Ports P2-P4(P0)

The Z8800 and Z8801 microprocessors can use port P0 for addressable I/O or external memory, where the address range is selectable at 0000H-0FFFH or 0000H-FFFFH depending on address pin definition.

2. Internal program memory (0/4/8/16 Kbytes) is located in emulation memory on the CEP board (two 6264 RAMs at U52 and U53). The size of the internal memory to be emulated is selected by placing jumpers on the personality board as follows.

Internal Memory Size	Jumper Setting			
	X10	X11	X12	X13
none (ROMless)	C	O	C	O
4K	O	C	O	C
8K	C	O	O	C
16K	O	C	C	O

C: Close
O: Open

Size of Internal Program Memory

3. If the user's program is larger than internal program memory or external data memory is used, then a SUEM/HUEM board must be used for emulation memory. To select SUEM/HUEM, set mode register P0M (F0H, BANK0) and PM (F1H, BANK0) with the R command, or with the user's program, to define Port 0 in address mode (P0=ADDR) and Port 1 in address/data mode (P1=A/D); i.e. set D7-D0=1 in register P0M (RS0F0=FF), and D5=1 in register PM (RS0F1=20).

   Example: Set RS0F0=FF and RS0F1=20 to specify -

   a) P00-P07=A8-A15
   b) P10-P17=AD0-AD7

4. However, if memory space is segregated, then input the EX command (Chapter 6), and logically select program memory and data memory using Data Memory select output. Define Port3-pin5 for Data Memory output, by setting mode register PM (F1H, BANK0) with the R command, or with the user's program, to P35=$\overline{DM}$. External segregated emulation memory is located in SUEM/HUEM, with program memory at 0000H-FFFFH and data memory at 11000H-1FFFFH.

Example: If the external program memory range is 4000H-BFFFH and data memory range is 1000H-8FFFH, then set emulation memory as indicated below for SUEM or HUEM.

SUEM a) Emulate program memory only by setting -
* memory segment to 0H
* starting address to 4000H

b) Or emulate data memory only by setting -
* memory segment to 10000H
* starting address to 1000H

HUEM Enable all emulation memory by setting -
* U14 and U36 S1-S9 ON and S10 OFF
Emulate program memory in bank 1 by setting -
* memory segment to 0H (U28S1-8 all ON)
Emulate data memory in bank 2 by setting -
* memory segment to 10000H (U43 S1 OFF, all others ON)

5. SUEM/HUEM can cover the entire emulation memory range required for external program and external data memory (max 128K bytes).

Because internal program memory is located on the CEP board at U52/U53, do not set the address for SUEM within the internal memory range. However, the offset address **cannot** be defined for HUEM. Therefore, the first 0/4/8/16 Kbytes of the memory bank containing program memory are disabled when emulation memory is segregated, and the first 0/4/8/16 Kbytes of both memory banks are disabled when emulation memory is combined. For information on HUEM switch settings refer to section 2.5.

Emulation Memory Selection

Required Memory Size / Emulation Memory Location	0,<4/8/16 Kbytes* (Microcontroller)	>0/4/8/16 Kbytes* (Microcomputer)	
		Segregated (set EX, & P35=$\overline{DM}$)	Combined (set DX)
SUEM	Not required	Use for program or data memory (max 128K bytes)	Max 64K bytes
HUEM	Not required	Use for program or data memory, or both (max 128K bytes)	Max 64K bytes
CEP U52/53(6264x2)**	Used as internal program memory (Max 16K bytes)		

* Depends on the target processor under emulation.

** If internal program memory is located in target ROM, either read the data into the host computer and then download it to MICE emulation memory, or place the target ROM (if a 2764) on the CEP-ZS8 at U52 or U53.

a) If HUEM is selected and memory is combined, then both program and data memory must be located in the same memory bank (HUEM has two separate 64K byte memory banks) with the memory segment set to OH (U37/U43 S1-8 all ON). Because the internal memory range is always disabled (0/4/8/16 Kbytes internal program memory is located on the CEP-ZS8), the maximum addressable memory on HUEM is 64 Kbytes.

However, if memory is segregated, then program memory should be located in one memory bank and data memory in the other bank. The maximum addressable memory on HUEM is then 128K bytes.

b) When using SUEM, up to 128K bytes of external memory can be defined. The start address can be specified anywhere within the available emulation memory (SUEM section).

2.7 RS-232C Cable Connection

After selecting the proper data rate and transmission characteristics, next determine whether the controlling device has a data terminal equipment (DTE) interface or a data communication equipment (DCE) interface, with or without handshaking. Display terminals are usually equipped with DTE interface; computer systems usually have both.

There are several methods for determining the type of interface on the controlling device. The first method is by simple trial and error. If this fails, procedures two and three listed below can be used to determine the interface type. (It must be known beforehand whether or not handshaking is selected as a software option for the controlling device, as it cannot be easily detected by examining hardware signals.) Subsequent instructions detail how to accomplish interface header rewiring if required.

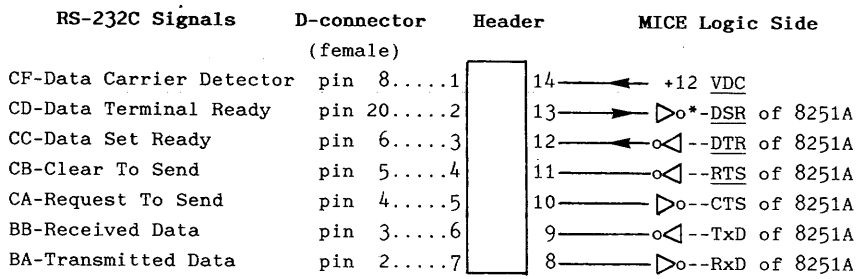
1. Try connecting the two devices together. If the response is correct, the interfaces match and no further adjustment is required. If the interconnect does not work, a mismatch exists and indicates that a connection change is necessary.

2. Data are transmitted on pin 2 and received on pin 3 of the D-connector for DTE devices; the reverse is true for DCE devices.

3. When not transmitting, the voltage (with respect to pin 7 [ground] of the D-connector) is -12 VDC on pin 2 for DTE devices and -12 VDC on pin 3 for DCE devices.

The header designated as U13 on the CEP board is used to configure the interface between NEW MICE-II and the controlling device. NEW MICE-II are shipped from the factory with a straight-wired header for DTE with handshaking.

Pins 1-7 on the interface header are connected to the female D-connector (J2) while pins 8-14 are connected to the NEW MICE-II, RS-232C interface logic. Pin definitions on the header are shown below:



*RS-232C interface gate for 1488 or 1489 depending on signal Input or Output.

The wiring configuration for all three header types is shown below. Note that if the controlling device has a DCE interface, incoming signals are now outgoing, and vice versa, on the same pins. By removing the cover to the header, the interface type used by MICE can be determined according the following wiring configuration.

DTE with handshaking	DCE with handshaking	DTE without handshaking	DCE without handshaking		
1 — 14	1 — 14	1 — 14	1 — 14	(from MICE)	Data Carrier Detector
2 — 13	2 X 13	2 | 13	2 [13	(to MICE)	Data Terminal Ready
3 — 12	3 X 12	3 | 12	3 [12	(from MICE)	Data Set Ready
4 — 11	4 X 11	4 | 11	4 [11	(from MICE)	Clear to Send
5 — 10	5 X 10	5 [10	5 [10	(to MICE)	Request to Send
6 — 9	6 X 9	6 — 9	6 X 9	(from MICE)	Received Data
7 — 8	7 X 8	7 — 8	7 X 8	(to MICE)	Transmitted Data

When wired for DTE, connect NEW MICE-II using a straight-wire cable with male D-connectors; for other header types, the cable must be rewired accordingly. If a display terminal is connected after making the proper connections, NEW MICE-II should respond when power is applied. (A display terminal should always be used to check new NEW MICE-II units to ensure that they are properly functioning.)

Finally, NEW MICE-II requires that both Request to Send and Data Terminal Ready inputs, pins 10 and 13 respectively on the logic side of the header, be at +12 VDC before it transmits any data. If the controlling device does not supply the necessary voltage, it can be obtained by reconnecting pins 10 and 13 directly to pin 14 (Received Line Signal Detector) of header U13 which is always at +12 VDC.

To accommodate computer systems which are sending commands and data too fast for NEW MICE-II, the Data Set Ready output, normally at +12 VDC, is pulled low by NEW MICE-II to -12 VDC. The signal is restored to +12 VDC when NEW MICE-II is again ready. Also, to accommodate display terminals with slow carriage-returns or line-feeds, six null characters are always transmitted after a carriage-return is issued.

2.8 Applying Power to NEW MICE-II

Connect the RS-232C cable from the controlling device to the female D-connector at the rear of NEW MICE-II. (The recommended power-on sequence is to first apply power to the target and then to NEW MICE-II; and to use the opposite sequence when powering off.) With the controlling device ready, connect the power cable with the locking tab pointing up, and apply power. Within a few seconds, the following start-up message should display:

```
***MICE-II-type V#.#***
>
```

type identifies target processor being emulated by the personality card.

#. # indicates version number of controlling program on personality card.

> is the prompt character indicating that NEW MICE-II is ready for a command.

In response to the command prompt character ">" enter a question mark "?", immediately followed by a carriage

return. If the data bits and parity are correctly matched, the NEW MICE-II command summary is listed; otherwise, the error message "**WHAT?**" is printed. Reset the switch section where necessary and remember to wait a few seconds before power is turned on again.

Notes 1. If target VCC is not provided when NEW MICE-II is powered up, the following message will display:

```
Z80    - NO TARGET VCC; NMI, INTR, RESET, BUSRQ DISABLED!
NSC800 - NO TARGET VCC; NMI, INTR, BREQ DISABLED!
Z8     - NO TARGET VCC; IRQ, RESET DISABLED, MEMORY COMBINED!
ZS8    - NO TARGET VCC; RESET DISABLED, MEMORY COMBINED!
```

2. The CPU is automatically reset after power-up, software reset command "r" (section 2.9) or MICE Reset command X (section 5.7). If the target system has any peripheral devices or slave processors which must be synchronized with a CPU reset, then the reset must be performed on the target side. A target reset signal will also reset the emulation CPU. However, the reset control signal for MICE must be enabled (section 6.2) for the Z80/Z8/ZS8, otherwise the target reset signal cannot reset the emulation CPU.

   Note that a synchronized reset may be performed by either MICE or the target system for the NSC800, since this processor supports a RESET output signal when the reset line for peripheral devices or slave processors on the target side is connected to reset pin-37.

2.8.1 No Response

If there is no response from NEW MICE-II, check the following items:

1. Check the RS-232C cable connection at both ends.

2. Check the power supply connections and voltages.

3. Check that the header has the proper interface.

4. Check that both Request to Send and Data Terminal Ready, pins 10 and 13 on the header are at +12 VDC.

5. Check that the controlling device has the proper voltage level requirements on its RS-232C inputs for transmission and reception.

6. Check that the baud rates of the controlling device and NEW MICE-II are the same, resetting if necessary. If a message does appear but is garbled, any combination of the baud rate, data length or parity could be incorrectly set.

7. Check the RS-232C cable for incorrectly wired or loose pins.

8. If a computer system is the controlling device, check that the driver program is running and the RS-232C cable is connected to the correct port and that the port is working.

Note: When changing the communication configuration switch, note that it is only read during a power-up or reset. After turning the power off, wait a few seconds before the power is turned on again to allow the capacitors to fully discharge.

If the problem still cannot be found, contact your local Microtek representative for further assistance.

2.8.2 Failure Device

During the delay prior to the start-up message the RAMs and EPROMs on the CEP board are tested. If any component failures exist, they are listed as detected in the following format: "**U## - FAILURE**", where ## is the component number on the personality board.

2.9 Control Processor Software Reset Command

The character "r" is the command to reset NEW MICE-II. Remember that except for this command all alphabetic characters must be entered in upper-case; no other lower-case characters are recognized.

CHAPTER 3

NEW MICE-II COMMAND LANGUAGE

All NEW MICE-II products have a common set of commands identified by a single or double character regardless of the processor being emulated. No additional time or effort is required to learn a new command language when the target processor changes.

These commands are described in the following chapters along with a variety of available options, though not all options are applicable for the different types of processors. For the specific processor being emulated, consult the Help "?" command. The commands described in the following chapters are grouped as follows:

NEW MICE-II Utility Commands

? Help Command
! Attention Command

Memory, Port and Register Commands

M Memory Display/Examine/Modify/Fill/Search Command
T Memory Checksum/Test/Transfer/Compare Command
A Line Assembly Command
Z Disassembly Command
I Port Input Command
O Port Output Command
R Register Display/Modify Command
J Jump/Branch Command
X Reset/Initialization Command

Control Signal Commands

D Disable/Display Control Signal Command
E Enable/Display Control Signal Command

Emulation and Trace Control Commands

G Go/Execution Command
H Halt/Breakpoint Set Command
F Forward Trace Command
B Backward Trace Command
L List Trace Buffer Command

Stepped Emulation Commands

C Single Cycle, Step Command
S Instruction Step Command

Utility Commands Involving a System

: Download Command (Intel Format)
/ Download Command (Tektronix Format)
U Upload Command

3.1 Command Syntax

NEW MICE-II indicates that it is ready to accept a command line by printing a greater-than character ">" on a new line. A command may then be entered and must be terminated by a carriage-return <CR>. The general syntax of NEW MICE-II commands is:

command [parameters] <CR>

where: **command** is the command representation.

parameters are one or more variable data supplied with the command. Parameters are alphanumeric; when a numeric parameter is called for, it must be entered in hexadecimal.

Where a space is shown in the syntax, either a space or comma can be used. A <CR> must be used to terminate a command input line. In most cases line feed <LF> or <CR> have the same effect, except where otherwise noted. Note that brackets [ ] and braces { } are used for describing command syntax only, and are not used in command input.

3.2 Notations and Conventions

A set of conventions is used to describe the structure of commands. The notations and rules are as follows:

1. An upper-case entry must be input.

2. A lower-case entry in the description of a command is the class-name for a parameter. A particular value for this class must be entered. A classname never appears in an actual operable command. For example, the lower-case entry - **"start-address"** means that NEW MICE-II will only accept a hexadecimal value as an address in the target processor's memory space.

3. A required entry is shown without any enclosures; whereas an optional entry is denoted by enclosing it in brackets. For example, in the command description - **"G[address]"**, the command "G" is required, and the brackets around the entry "address" means that it is optional in this command.

   Where brackets are within another set of brackets, the entry enclosed by the inner brackets may only be entered if the items outside those inner brackets are first entered. For example, in the command description - "I **port[count[time]]**", th
   is required; and the brackets mean tha
   tion of "count" and "time" is optional
   mand. However, a "count" must first be en
   "time" value is to be specified.

4. Where an entry must be selected from a choice of two or more, the choices for the required entry are enclosed in braces and separated by vertical bars. For example - "**{S|M|a3[V]}**" indicates that either "S", "M" or "a3" must be entered.

5. Where a choice exists for an optional entry, the choices are enclosed in brackets and separated by vertical bars. For example, "**[I|H|T]**" indicates that either "I", "H" or "T" may be entered.

6. "Addx" is a hex address with a wildcard byte pair of "XX" or "XXXX", or a wildcard nibble of "X" [only for the third digit (e.g. X23) for 8048]; where X indicates that the digits are "don't care". For example - "**12XX**" means all addresses between 1200H and 12FFH.

7. Commands A/C/S/I/R all use carriage-return <CR> or line-feed <LF> to display the next line.

8. In the Memory Modify (M) command, use a <CR> to advance to the next location and a <LF> to return to the previous location.

9. Entries underlined in command examples indicate user input.

3.3 Editing Characters

Each character entered on the keyboard is stored in a line editing buffer until <CR> is entered. If more than 80 characters are entered without inputting a <CR>, an error message is printed, and the command is ignored.

The line editing buffer can be edited or entirely deleted by using special non-printable editing characters. Control characters are entered by holding down the control key <CTRL> while the character is typed. Control character input is expressed with the control command enclosed in < > brackets.

BACKSPACE deletes the preceding character from the line buffer and from the display. Repeated usage is allowed. <CTRL-H> performs the same function. When a hardcopy terminal is used instead of a display screen, RUBOUT should be used.

RUBOUT deletes the preceding character from the line buffer and echoes the deleted character on the display, preceded by a backslash character. Repeated usage is allowed. On some terminals, this key may be labeled as DELETE or DEL.

ESCAPE ignores the current contents of the line buffer and prompts ">" for a new command on the next line. This key is also used to terminate commands in process and to return to the prompt state. ESCAPE is also expressed as <ESC>. On some terminals, this key may be labeled ESC. <CTRL-Y> performs the

same function.

<CTRL-R> causes a <CR> or <LF>, followed by a redisplaying of the current undeleted contents in the line buffer. This is useful to see a clean copy of the command line after RUBOUT has been used.

<CTRL-X> ignores the current contents of the line buffer and shifts the cursor to the first position of the next line, awaiting input of new data.

3.4 Control Characters and Delimiters

The following control characters have special meaning for all NEW MICE-II firmware:

<CTRL-J>: is the same as <LF>.
<CTRL-M>: is the same as <CR>.
<CTRL-S>: stops data transmission from NEW MICE-II.
<CTRL-Q>: continues data transmission from NEW MICE-II.
<CTRL-Y>: is the same as <ESC>.

3.5 Reference Program for Examples

The following sample program for a Z80 target is used as the basis for most of the examples listed in this user's manual. It includes commonly used instructions for performing various bus cycles.

```
LOC     OBJ       LINE   LABEL    SOURCE CODE
0000    310020    0001   B0000    LD    SP,2000
0003    CD0001    0002            CALL  0100
0006    DD210030  0003            LD    IX,3000
000A    110010    0004            LD    DE,1000
000D    1A        0005   B000D    LD    A,(DE)
000E    DD7700    0006            LD    (IX+00),A
0011    DD23      0007            INC   IX
0013    13        0008            INC   DE
0014    7B        0009            LD    A,E
0015    FE0F      0010            CP    0F
0017    C20D00    0011            JP    NZ,000D
001A    C30000    0012            JP    0000
0100    110010    0001            LD    DE,1000
0103    3E20      0002   B0103    LD    A,20
0105    12        0003            LD    (DE),A
0106    7B        0004            LD    A,E
0107    13        0005            INC   DE
0108    FE0F      0006            CP    0F
010A    C20301    0007            JP    NZ,0103
010D    C9        0008            RET
010E    3A003A    0009            LD    A,(3A00)
```

CHAPTER 4

NEW MICE-II UTILITY COMMANDS

These commands allow the user to query NEW MICE-II for a summary of the commands and syntax that are available for the target under emulation. In addition, the user can query to display the processor type being used.

Notes: 1. If any code other than FFH*1 is placed in firmware at the location indicated in table 4-1, it will cause the handshaking code at this location to be sent to the host computer (or terminal) when MICE is waiting for input. This extra code can be used to improve interface efficiency with the host.

2. The handshaking code 03H*2 is sent to the host computer when MICE is waiting for input. Any code other than 003H may be placed in firmware (table 4-1) to suit the user's specific requirements.

NEW MICE-II	CEP Location	Address	Handshaking Code	
			FFH *1	03H *2 (ETX)
Z80	U4	1FFDH	V1.0-V3.0	V3.1 & later
NSC800	U4	1FFDH	V1.0-V3.1	V3.2 & later
Z8	U1	3FFDH	V3.0	V3.1 & later
ZS8	U1	3FFDH		V3.0 & later

Table 4-1 Handshaking Codes

4.1 Help Command - ?

?

? is the command for Help.

The command summary or the target processor currently being emulated is displayed on the terminal. All commands are common regardless of target processor type, but command parameters may differ for the different processors under emulation.

Example: Display the command summary for NEW MICE-II Z80.

```
>?
ASSEMBLY            A [loc]
BACKWARD TRACE      B [R] addx[ c[ q]]
CYCLE STEP          C [c]
DISABLE             D [N|I|B|R]
ENABLE              E [N|I|B|R]
FORWARD TRACE       F [R] addx[ c[ q]]
EXECUTION           G [address]
BREAKPOINT          H [0[1|2]|1 [addx[ c[ q]]]|2 [addr]]
INPUT               I port[ c[ time]]
JUMP                J address
LIST TRACE          L [step[ a1[ a2[ q...]]]|S[step]|Z[step]|N]
MEMORY              M [a1[ a2[ d1...[ d8][ S]]]]
OUTPUT              O port d1[ d2[...[ d8]]]
REGISTER            R [P][A|B|C|D|E|H|L|I|F|X|Y|S]
INSTRUCTION STEP    S [S|R][c]|[Z]
TRANSFER/TEST       T a1 a2 S|M|a3[ V]
UPLOAD              U a1 a2 [ T| I]
RESET               X
DISASSEMBLY         Z [a1[ a2]]
DOWNLOAD            : (INT), / (TEK)
```

```
HELP                ? [B]
ATTENTION           !
>
```

The above summary does not fully follow the notations and conventions previously described to avoid a lengthy display.

NEW MICE-II Z80 is used as the basis for all examples in this manual. Command syntax varies for different NEW MICE-II emulators; for the proper syntax consult the Help (?) command.

4.2 Attention Command - !

!

! is the command for Attention.

The target processor type currently being emulated together with firmware version number by the personality card in NEW MICE-II is displayed on the terminal. Six characters are used for identification.

Example: Display the processor type currently being emulated (Z80).

```
>!
EPZ80R
>
```

The messages for 8-bit series processor types are:

NEW MICE-II	TARGET PROCESSOR TYPE	
Z80	(Zilog Corporation)	- EPZ80/EPZ80R
NSC800	(National Semiconductor)	- EP800
Z8	(Zilog Corporation)	- CEP-Z8
ZS8	(Zilog Corporation)	- EP-ZS8

CHAPTER 5

MEMORY, PORT AND REGISTER COMMANDS

These commands give access to the contents or current values stored in designated memory locations, I/O ports and registers. The memory type, maximum amount of addressable memory, and total amount of addressable I/O are shown below. Note that targets with only one memory type have no distinction between program and data memory. For microcontroller versions with multiple memory types, refer to section 2.6.

CPU	Addressable Memory	Addressable I/O
Z80	0000H-FFFFH	0000H-FFH
NSC800	0000H-FFFFH	00H-FFH

NEW MICE-II always checks that memory type and memory/port addresses are valid before an operation is performed. References to an invalid address result in an error message (**ERROR!**) being printed and the command ended. NEW MICE-II will verify memory write operations. However, it will not verify memory read or I/O operations. If there is no memory when executing a memory read operation, then data read is random. And if there is nothing connected to the port when executing an I/O operation, then data written is lost and data read is random.

Notes: 1. When performing READ/WRITE functions, if external circuitry does not respond with the following signal, **"TARGET IS NOT READY!!"** will appear on the console and the contents of the registers will be retained.

Z80	$\overline{\text{WAIT}}$
NSC800	$\overline{\text{WAIT}}$

However, if the target does not respond when performing READ/WRITE functions for NEW MICE-II Z8 or ZS8, incorrect data will be displayed.

2. A blank space (ASCII 20H) along with a handshaking code (ASCII 03H) is sent at the end of each displayed line immediately before the cursor for the Assembly, Memory Modify and Register Modify commands. This code serves as an end of data message to the host computer; and is provided to help in implementing symbolic debuggers. (Refer to page 4-1.)

5.1 Memory Display/Examine/Modify/Fill/Search Command - M

```
Z80/NSC800 - M[start-address[ end-address[ data-1[ data-2[ ...data-8]][ S||||
Z8/ZS8       M[|P|X][start-address[ end-address[ data-1|..data-8]][ S|||
```

M is the command for Memory Display/Examine/Modify/Fill/Search. If no other parameters are specified, inputting "M<CR>" will display the next 256 bytes starting at the current PC.

P|X are memory type designations for microcontrollers indicating program memory and external data memory respectively. (Refer to section 2.6 for a detailed description of memory types.)

start-address is a hexadecimal address of the emulation CPU indicating a memory location where the operation is to begin.

end-address is a hexadecimal address of the emulation CPU indicating the last memory location of the specified range.

data-1...8 are hexadecimal or ASCII data. Data in ASCII must be enclosed within two apostrophes (e.g. 'AR'). Where a "block fill" or "block search" operation is specified, the block may be up to 8 bytes in hex or 8 characters in ASCII. Combined use of hex and ASCII is permitted. If the address range (from start to end-address) is smaller than the block size (data1 - data8), then block-fill is still executed but excess trailing data is ignored; for block-search however, **"MEMORY SEARCH FAILURE!"** will display. (Note that the apostrophe ['], lower case "r" and MICE editing/control characters are not allowed for ASCII input.)

S defines a "search" option which looks for specified data.

Where the start and end-address are defined for a memory range in the emulation processor, the end-address must be greater than or equal to the start-address or an error message is printed and the command ends.

5.1.1 Memory Display

M[start-address[end-address]]

Input a start and end-address to display the content of a memory range. Memory contents are displayed in hexadecimal and ASCII. Note that data without an ASCII equivalent is indicated by a period. The end-address must be greater than or equal to the start-address or an error message is printed and the command ends. The display can be terminated by entering an <ESC>.

Example: Display program memory contents 0H to 16H.

```
>M0 16
      00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F     ASCII-CODE
0000 31 00 20 DD 21 31 00 CD 1B 00 DD 21 54 00 CD1B  1. .!1.....!T...
0010  00 DD 21 99 00 CD 1B                           ..!....
>
```

5.1.2 Memory Examine/Modify

M start-address

If only the start-address is specified, the content of that memory location is first displayed. NEW MICE-II then waits for input. To advance to the next memory address, a <CR> should be entered; the next memory location's content is then displayed, and NEW MICE-II again waits. To go back to the previous memory address a <LF> should be entered; the previous memory location's content is then displayed, and NEW MICE-II again waits. Entering a <CR> or <LF> repeats the entire sequence. If an <ESC> is entered, the command ends.

To change the memory content displayed, enter a new value and a <CR> or <LF>. (Note that the apostrophe ['], lower case "r" and MICE editing/control characters are not allowed for ASCII input.) After writing data to memory, verification of memory starts automatically. If any data does not match, the following message displays: **"MEMORY VERIFICATION FAILURE!"**.

Example: Examine and modify the memory contents from address location 1000H, then display the results.

```
>M 1000
 1000 00 31<CR>
 1001 00 'A'<CR>
 1002 00 <LF>
 1001 41 31<CR>
 1002 00 <ESC>
>M 1000 100F
      00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F     ASCII-CODE
1000 31 31 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0000  11..............
>
```

5.1.3 Memory Fill

M start-address end-address data-1[data-2[...data-8]]

Input a start and end-address for the memory range to be filled. A specified data value or data block may be written into the defined range.

Example: Write the ASCII value "NEW DATA" into the memory range 1017H to 102FH and then display the memory contents for the range 1000H to 102FH.

```
>M 1017 102F 'NEW DATA'
>M 1000 102F
      00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F     ASCII-CODE
1000 31 31 00 00 00 00 00 00 00 00 00 00 00 00 0000  11..............
1010  00 00 00 00 00 00 00 4E 45 57 20 44 41 54 41 4E  .......NEW DATAN
1020  45 57 20 44 41 54 41 4E 45 57 20 44 41 54 41 4E  EW DATANEW DATAN
>
```

5.1.4 Memory Search

M start-address end-address data-1[data-2[...data-8]] S

Input a start and end-address for the memory range to be searched. A specified data value or data block may be searched for within the defined range.

Example: Search for string "BA 10" in the range 0H to 1FH, which is not found; then search for string 'NEW' in the same address range.

```
>M 1000 102F
     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F     ASCII-CODE
1000 31 31 00 00 00 00 00 00 00 00 00 00 00 00 0000  11..............
1010  00 00 00 00 00 00 00 4E 45 57 20 44 41 54 41 4E  .......NEW DATAN
1020  45 57 20 44 41 54 41 4E 45 57 20 44 41 54 41 4E  EW DATANEW DATAN
>M 1000 102F BA 10 S
 MEMORY SEARCH FAILURE!
>M 1000 102F 'NEW' S
 MEMORY MATCH AT ADDRESS 1017
>
```

5.2 Memory Checksum/Test/Transfer/Compare Command - T

```
Z80/NSC800 - T start-address end-address {S|M|address-3[ V]}
Z8/ZS8     - T[P|X]start-address end-address {S|M|address-3[ V]}
```

T	is the command for Memory Checksum/Test/Transfer/Compare.
P|X	are memory type designations for microcontrollers indicating program memory and external data memory respectively. (Refer to section 2.6 for a detailed description of memory types.)
start-address	is a hexadecimal address of the emulation CPU indicating a memory location where the operation is to begin.
end-address	is a hexadecimal address of the emulation CPU indicating the last memory location of the specified range.
S	displays the checksum of the contents in the specified range.
M	performs a memory test for the specified range.
address-3	is a hexadecimal address in the emulation CPU specifying either a destination address, where data is to be transferred, or the start of a second memory range used in "block compare" (section 5.2.4).

V specifies "block compare".

Define the start and end-address for a memory range in the emulation processor. The end-address must be greater than or equal to the start-address or an error message is printed and the command ends.

5.2.1 Memory Checksum

T start-address end-address S

An "S" following the range specification causes the checksum to be displayed. The checksum is calculated by taking the hexadecimal sum of the contents for the indicated range, with carry added back, modulo 256.

Example: First display the program memory contents for the range 1025H-102AH, and then calculate the checksum for the same range.

```
>M 1025 102A
      00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F    ASCII-CODE
1020                  54 41 4E 45 57 20                    TANEW
>T 1025 102A S
THE CHECKSUM IS: A0
>
```

5.2.2 Memory Test

T start-address end-address M

An "M" following the range specification causes a memory test to be performed on the indicated range. The upper-byte and lower-byte of the address whose memory is to be tested are exclusive-or'ed (XOR) and written into memory for the entire range. The data are verified and then their complements are written into the entire range and thoroughly checked again. If the comparison fails in either pass, the test is stopped at the failed address and that address is displayed. The original contents in memory are destroyed in this test.

Example: Test the memory range 1019H through 1023H, displaying the memory contents for the range 1010H to 102FH both before and after.

```
>M 1010 102F
      00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F     ASCII-CODE
1010  00 00 00 00 00 00 00 4E 45 57 20 44 41 54 41 4E  .......NEW DATAN
1020  45 57 20 44 41 54 41 4E 45 57 20 44 41 54 41 4E  EW DATANEW DATAN
>T 1019 1023 M
 RAM OK!
>M 1010 102F
      00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F     ASCII-CODE
1010  00 00 00 00 00 00 00 4E 45 F6 F5 F4 F3 F2 F1 F0  .......NE.......
1020  CF CE CD CC 41 54 41 4E 45 57 20 44 41 54 41 4E  ....ATANEW DATAN
>
```

Example: Test the memory range 3FF0H through 400FH, where memory is enabled only through 3FFFH.

```
>T3FF0 400F M
ADDRESS (4000) RAM ERROR!
>M3FF0 400F
      00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F     ASCII-CODE
3FF0  30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F  0123456789:;<=>?
4000  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ................
>
```

5.2.3 Memory Transfer

T start-address end-address address-3

An address-3 (dest-address) specification indicates that the memory content in the defined range is to be transferred into memory beginning at address-3. Data is transferred, one location at a time beginning at the start address of the destination range. If the memory ranges defined in the transfer command overlap, the transfer will begin at the end address of the destination range to prevent overwrite.

Example: First display the memory content for the range 1000H to 102FH, then transfer the memory content from the range 2000H-202FH to memory beginning at 1000H. Finally, display the results of the transfer.

```
>M 2000 202F
      00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F     ASCII-CODE
2000 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0000  ................
2010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
2020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
>T 1000 102F 2000
>M 2000 202F
      00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F     ASCII-CODE
2000  31 31 00 00 00 00 00 00 00 00 00 00 00 00 00 00  11..............
2010  00 00 00 00 00 00 00 4E 45 F6 F5 F4 F3 F2 F1 F0  .......NE.......
2020  CF CE CD CC 41 54 41 4E 45 57 20 44 41 54 41 4E  ....ATANEW DATAN
>
```

5.2.4 Memory Compare

T start-address end-address address-3 V

A "V" following the address-3 specification executes block compare for the indicated memory ranges. The block from start-address to end-address is compared with the block beginning at address-3. If comparison is successful, ↓**"COMPARISON OK!"** will display; otherwise the failure addresses along with the incorrect data are displayed.

Example: First display the memory contents from 1000H to 102FH and 2000H-202FH, then compare the data block.

```
>M 1000 102F
      00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F     ASCII-CODE
1000 31 31 00 00 00 00 00 00 00 00 00 00 00 00 0000  11..............
1010  00 00 00 00 00 00 00 4E 45 F6 F5 F4 F3 F2 F1 F0  .......NE.......
1020  CF CE CD CC 41 54 41 4E 45 57 20 44 41 54 41 4E  ....ATANEW DATAN
>M 2000 202F
      00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F     ASCII-CODE
2000 31 31 00 00 00 00 00 00 00 00 00 00 00 00 0000  11..............
2010  00 00 00 00 00 00 00 4E 45 F6 F5 F4 F3 F2 F1 F0  .......NE.......
2020  CF CE CD CC 41 54 41 4E 45 57 20 44 41 54 41 4E  ....ATANEW DATAN
>T 1000 102F 2000 V
 COMPARISON OK!
>
```

Example: Compare the data block from 1000H to 102FH
with the address range beginning at 0H.

```
>T 1000 102F 0 V
(1001)=31   ;(block 1 failure address) = data 1
(0001)=00   ;(block 2 failure address) = data 2
>
```

5.3 Line Assembly Command - A

A[start-address]

A	is the command for Assembly. If no other parameters are specified, inputting "A<CR>" will execute assembly beginning at the current PC.
start-address	is the hexadecimal address in the emulation processor's program memory where MICE begins storing the entered assembly language program. Note that only emulation program memory may be specified or the Z8 and ZS8.)

Rather than enter programs or program changes in machine code using the previously described memory (M) command, the NEW MICE-II resident assembler accepts and converts mnemonic inputs into machine code. The converted code is then stored in the emulation processor's program memory starting at the indicated start-address. The assembler does not recognize symbolic labels or constants other than hexadecimal values. The instruction mnemonics accepted are those adopted by the original manufacturer for the processor being emulated. In addition to these mnemonic codes (listed in the appendices), the following three instructions are also supported during assembly.

DB - Define Byte	(1-6 bytes)
DW - Define Word	(1 word)
DS - Define Storage	(0H-FFFFH)

DB accepts both hex and ASCII codes, while DW and DS are restricted to hex data. (Note that the apostrophe ['], lower case "r" and MICE editing/control characters are not allowed for ASCII input.) If more than 6 bytes are keyed in for DB, excess data is truncated on the right and NEW MICE-II displays: "**WARNING: ONLY 6 BYTES ARE VALID!**". If more than 1 word (4 digits) are keyed in for DW or DS, the input data is ignored and "**ERROR CODE, TRY AGAIN!**" is displayed.

After inputting the assembly command, NEW MICE-II displays the following column headings:

```
>AO
LOC      OBJ    LINE      LABEL          SOURCE CODE
0000            0001                     *
```

where the decimal value of 0001 under the column "LINE" indicates the line number being entered and the "*" indicates the new cursor position. NEW MICE-II then waits for an assembly language line input from the user. The source code column consists of an opcode and operand, and must be terminated with either a <CR> or <LF>. NEW MICE-II assembles the line and stores the machine code beginning at the indicated start-address. The code for the program memory location to be stored is printed under the column "OBJ". The next line number is printed and NEW MICE-II again waits. If an <ESC> is entered instead, the command ends.
NEW MICE-II performs data verification after executing a memory WRITE. If any data does not match, the following message displays: "**MEMORY WRITE FAILURE!**". If an invalid program instruction is entered, NEW MICE-II displays: "**ERROR CODE, TRY AGAIN!**".

Example: Enter a simple program starting at location 0H.

```
>A0
LOC     OBJ        LINE    LABEL    SOURCE CODE
0000    310020     0001             LD SP,2000 <CR>
0003    CD0001     0002             CALL 100 <CR>
0006    DD210030   0003             LD IX,3000 <CR>
000A    110010     0004             LD DE,1000 <CR>
000D    1A         0005             LD A,(DE) <CR>
000E    DD7700     0006             LD (IX+0),A <CR>
0011    DD23       0007             INC IX <CR>
0013    13         0008             INC DE <CR>
0014    7B         0009             LD A,E <CR>
0015    FE0F       0010             CP 0F <CR>
0017    C20D00     0011             JP NZ,0D <CR>
001A    C30000     0012             JP 0 <CR>
001D               0013             <ESC>
>

>A100
LOC     OBJ        LINE    LABEL    SOURCE CODE
0100    110010     0001             LD DE,1000 <CR>
0103    3E20       0002             LD A,20 <CR>
0105    12         0003             LD (DE),A <CR>
0106    7B         0004             LD A,E <CR>
0107    13         0005             INC DE <CR>
0108    FE0F       0006             CP 0F <CR>
010A    C20301     0007             JP NZ,103 <CR>
010D               0008             <ESC>
>
```

Example: An error occurs when an incorrect mnemonic is entered. In this case, non-hexadecimal data cannot be used in the data field.

```
>A E
LOC      OBJ              LINE    LABEL     SOURCE CODE
000E                      0011              LD A(IX+00)
ERROR CODE, TRY AGAIN!
000E     DD7E00           0011              LD A,(IX+00)
0011                      0012              <ESC>
>
```

When making changes in an existing program, it is advisable to check the program around the modified area using the Disassembly command (section 5.4) before and after the change. Rechecking the modified area is important to assure that new program changes do not affect the surrounding program.

5.4 Disassembly Command - Z

Z[start-address[end-address]]

Z is the command for Disassembly. If no other parameters are specified, inputting "Z<CR>" will execute disassembly for the next 16 lines starting at the current PC.

start-address is the hexadecimal address in emulation processor's program memory where display of disassembled memory content begins. (Remember that only emulation program memory may be specified for the Z8 and ZS8.)

end-address is the hexadecimal address in the emulation processor's program memory indicating the last memory location of the range to be disassembled and displayed. (Remember that only emulation program memory may be specified for the Z8 and ZS8.)

If only the start-address is specified, the content for that memory location is disassembled and displayed; more data are then read from subsequent locations to complete the instruction if necessary. An end-address specification causes the memory contents in the defined memory range to be disassembled and displayed. The end-address must be greater than or equal to the start-address or an error message is printed (**INPUT ADDRESS ERROR**) and the command ends. If an illegal machine code is encountered, disassembly terminates at the illegal

code's address.

NEW MICE-II uses a two-pass disassembler with all branch and subroutine call addresses first identified and converted to labels; a total of 900 labels can be stored for disassembly. Depending on the range to be disassembled, there may be a pause before any data is displayed.

Example: Disassemble the previously entered program. Note that the byte following the end-address is also read in order to complete the instruction for this example. Labels include a prefix (B for branch or S for subroutine) and address.

```
>Z0 1A
LOC     OBJ              LINE    LABEL    SOURCE CODE
0000    310020           0001    B0000    LD    SP,2000
0003    CD0001           0002             CALL  0100
0006    DD210030         0003             LD    IX,3000
000A    110010           0004             LD    DE,1000
000D    1A               0005    B000D    LD    A,(DE)
DD7700          0006              LD    (IX+00),A
0011    DD23             0007             INC   IX
0013    13               0008             INC   DE
0014    7B               0009             LD    A,E
0015    FE0F             0010             CP    0F
0017    C20D00           0011             JP    NZ,000D
001A    C30000           0012             JP    0000
     DISASSEMBLY COMPLETED
>
```

Example: If an illegal machine code is encountered, disassembly terminates at the illegal code's address.

```
>Z1E 30
LOC      OBJ              LINE     LABEL     SOURCE CODE
001E     110010           0010               LD    DE,1000
0021     DDEE00FE         0011     B0021
            ERROR CODE!
>
```

5.5 Port Input Command - I

I port[count[duration]]

I — is the command for Port Input.

port — is the hexadecimal address of the emulation processor's input port that is to be read and displayed. Note that only port 0-3 (0H-3H) can be used for NEW MICE-II Z8 and only port 0-4 (D0H-D4H) can be used for NEW MICE-II ZS8.

count — is a hexadecimal value from 00H to FFH specifying the number of times the input port is to be read, with 00H indicating 256 times.

duration — is a hexadecimal value from 00H to FFH specifying the interval in milliseconds between each read, with 00H indicating 256 milliseconds.

The input port command has two modes of operation. If neither the count nor the duration is specified, a range of port contents can be read and displayed beginning with the indicated port address.

NEW MICE-II first reads and displays the contents of the port specified and waits for input from the user. To advance to the next port, enter either a <CR> or <LF>; the next port's content is then read and displayed, and NEW MICE-II again waits. Entering a <CR> or <LF> repeats the entire sequence.

Example: Read and display the contents of ports 80, 81 and 82.

```
>I80
 80 DC<CR>
 81 87<LF>
 82 FF<ESC>
>
```

Specifying a **count** with or without a duration interval, NEW MICE-II first reads from the specified input port the number of times indicated and then displays the contents.

Duration defines the interval between each read in milliseconds, permitting compensation for data inputs which are time critical. If no interval is specified, then data is read at one millisecond intervals.

Example: Read and display the next 64 (40H) values for port D0H at 10 (0AH) millisecond intervals.

```
>ID0 40 A
PORT 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F     ASCII-CODE
D0   36 22 AE 9B 2B 18 C7 77 26 D6 86 35 E5 95 44 F4  6"..+..w&..5..D.
     A3 53 03 B2 62 12 C1 71 20 D0 80 2F DF 8E 3E EE  .S..b..q ../..>.
     9D RD FD AC 5C 0B BB 6B 1A CA 7A 29 D9 88 38 E8  .M..\..k..z)..8.
     97 47 F6 A6 56 05 B5 65 14 C4 73 23 D3 82 32 E0  .G..V..e..s#..2.
>
```

Note that the interval can be up to 256 milliseconds (approximately a quarter of a second). If the number of times the port to be read is also 256, the elapse time before any data displayed is approximately 65 seconds. Therefore, to end the command before display begins, enter an <ESC>.

5.6 Port Output Command - O

O port data-1[data-2[...data-8]]

O is the command for Port Output.

port is a hexadecimal address of the emulation processor's output port. Note that only port 0-3 (0H-3H) can be used for NEW MICE-II Z8 and only port 0-4 (D0H-D4H) can be used for NEW MICE-II ZS8.

data-1...8 are hexadecimal values to be written into the specified output port of the emulation processor in sequential order.

If only one value is specified, it is written to the indicated output port. If more than one value is specified, each value is written to the indicated port at one millisecond intervals with the first value going out immediately.

Example: Write the value 48H to port 01H.

```
>O 1 48
>
```

Example: Write values 1, 2 and 3 to port 05H. Value 1 is written immediately followed by value 2 after one millisecond and value 3 after another millisecond.

```
>O 5 1 2 3
>
```

5.7 Reset/Initialization Command - X

X[address]

X is the command for Reset/Initialization.

address is a hexadecimal addresses in the emulation CPU's program memory where emulation is to begin.

A signal pulse is generated on the appropriate pin of the emulation processor to reset it. The processor's program counter, internal registers and flags are all altered. A start address can be defined which sets the emulation processor's program counter to the specified program memory address. Only the emulation processor is reset; any logic or peripherals connected to the target input reset pin are not affected.

In addition, control and interrupt signals to the emulation processor that are selectively controllable by the Disable and Enable commands are activated if NEW MICE-II is connected to a power-applied target system. However, if no target system is connected to NEW MICE-II, those signals will be disabled.

Example: Reset the emulation processor, showing changes in the program counter (PC) and registers before and after.

```
>R
   A  B  C  D  E  H  L  I  F    X    Y    S   PC
  FF F7 FE FD EF FB FF 00 FF FFFF FEFF FFFB 0010
>X
>R
   A  B  C  D  E  H  L  I  F    X    Y    S   PC
  FF F7 FE FD EF FB FF 00 FF FFFF FEFF FFFB 0000
>
```

5.8 Register Display/Modify Command - R

5.8.1 Register Command for NEW MICE-II Z80 and NSC800

R[register]

R is the command for Register Display/Modification.

register is an alphabetic character indicating the target register whose content is to be displayed or modified. The Z80 and NSC800 have two sets of registers that may be displayed or changed with the R command. Input "R" to select the current set or "RP" to select the alternate set. The register set that can be displayed and changed by the R command is listed in the following table; where alternates are available, they are listed under the mnemonic heading with an appostrophe.

Mnemonic	Register	Contents
A A'	Accumulator	8 bits
B B'	General Purpose	8 bits
C C'	General Purpose	8 bits
D D'	General Purpose	8 bits
E E'	General Purpose	8 bits
H H'	General Purpose	8 bits
L L'	General Purpose	8 bits
I	Interrupt Mask	8 bits
F F'	Flags	8 bits(6 flags)
X	Index IX	16 bits address
Y	Index IY	16 bits address
S	Stack Pointer	16 bits address

5.8.1.1 Display Registers

R

All register contents are displayed if no specific register is defined. Note that the displayed value of the program counter (PC) always indicates the address for the current instruction's opcode.

Example: Display register contents.

```
>R
 A  B  C  D  E  H  L  I  F    X    Y    S   PC
 FF F7 FE FD EF FB FF 00 FF FFFF FEFF FFFB 0000
>
```

5.8.1.2 Modify Registers

R register

If a register is specified, the content of that register is first displayed; NEW MICE-II then waits for input. To advance to the next register enter either a <CR> or <LF>; the next register's content is then displayed and NEW MICE-II again waits. Entering a <CR> or <LF> repeats the entire sequence unless the next register to display is "PC". The J command (section 5.9) must be used to change the program counter. If an <ESC> is entered, the command ends. To change the content of the displayed register, enter a new value in hex or ASCII (enclosed with two apostrophes, e.g. 'A') and <CR> or <LF>.

Example: Examine the specified register and change its content.

```
>RA
 A FF 02<CR>
 B F7 03<CR>
 C FE <ESC>
>R
  A  B  C  D  E  H  L  I  F    X    Y    S   PC
 02 03 FE FD EF FB FF 00 FF FFFF FEFF FFFB 0000
>
```

5.8.2 Register Command for NEW MICE-II Z8

R[P|G|W|a1[a2[d1[..d8]]]]

NEW MICE-II Z8 has three sets of registers:

R	displays all register contents.
RP	displays control or peripheral register contents.
RG	displays general purpose register contents (4H-7FH).
RW	displays working register contents (16 bytes within 0H-7FH where the start-address is 0H or a multiple of 10H). The pointer can be set by the user program or by modifying register "FD". If the register pointer is at an unused range (80H-E0H), working register contents will display as "--"

Ra1[a2[d1[..d8]] can display, modify or fill general purpose registers. Note that 4H≤a1≤a2≤7FH. See section 5.1 for more information on addresses (a1-a2) and data (d1-d8).

Example: Display all control and peripheral register contents. The current PC is shown as 000C. Note that "--" indicates write only registers.

```
>RP
 PC   SIO TMR T1 PR1 TO PRO P2M P3M P01 IPR IRQ IMR FLG RP SPH SPL
      F0  F1 F2  F3 F4  F5  F6  F7  F8  F9  FA  FB  FC FD  FE  FF
000C 00  50 00  -- FF  --  --  --  --  --  00  36  89 20  34  56
>
```

Example: Display all general purpose register contents. "WR" indicates working registers. Note that "**" indicates I/O registers.

```
>RG
     0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
R0   ** ** ** ** 01 01 01 01 01 01 01 01 01 01 01 01
R1   12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12
WR   0A 31 0C E3 00 BF 46 23 23 23 23 78 DA 0E 23 23
R3   34 34 34 34 34 34 34 34 34 34 34 34 34 34 34 34
R4   45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45
R5   56 56 56 56 56 56 56 56 56 56 56 56 56 56 56 56
R6   67 67 67 67 67 67 67 67 67 67 67 67 67 67 67 67
R7   78 78 78 78 78 78 78 78 78 78 78 78 78 78 78 78
>
```

Example: Display working register contents with the register pointer (RP) set to 20H.

```
>RW
RP   0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
20  0A 31 0C E3 00 BF 46 23 23 23 23 78 DA 0E 23 23
>
```

Example: Modify the contents of register "F8".

```
>RF8<CR>
RF8 FF 96<CR>
RF9 FF<ESC>
>
```

5.8.3 Register Command NEW MICE-II ZS8

R[Rn|Fflag]|G[a1[a2[d1[..d8]]]]|S[[0|1]a1]

R	displays working register contents (16 bytes within 00H-FFH). The pointer can be set by the user program or by modifying registers D7H/D6H. The contents of reserved registers and "write only" registers are displayed as "--", while the contents of port registers are displayed as "**".
RRn	permits modification of working registers.
RFflag	permits modification of flag register with binary input.
RG	displays general purpose register contents (00H-FFH).
RG a1[a2[d1[..d8]	can display, modify or fill general purpose registers.
RS	displays contents of special purpose registers, including system register (D0H-DFH), mode and control registers (E0H-FFH), and registers C0H-CFH (used as working registers only).

RS[0|1]a1 permits modification of special purpose registers, where [0|1] indicate bank selection.

See section 5.1 for more information on addresses (a1-a2) and data (d1-d8).

Example: Display working register contents with the register pointers RP0 and RP1 set to C0H and C8H respectively, and the PC set to 0020. Also note that the flags "CZSVDHFB" are displayed with binary data.

```
>R
RP0(C0)  R0=55 R1=AA R2=A2 R3=55 R4=75 R5=AA R6=AA R7=55  CZSVDHFB  PC=0020
RP1(C8)  R8=55 R9=FA RA=AA RB=55 RC=77 RD=AE RE=AB RF=75  01010000
>
```

Example: Modify working register R8.

```
>RR8<CR>
R8 55 AA<CR>
R9 FA <ESC>
>
```

Example: Modify the content of the flag with binary input (1 or 0).

```
>RFC
C 0 <CR>
Z 1 0<CR>
S 0 1<CR>
V 1 <ESC>
>
```

Example: Display all general purpose register contents.

```
>RG
     0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
RG0 55 A2 AA 15 57 BA AA D6 55 A8 AA 5C 55 BE AE F5
RG1 55 AA AA 55 55 AA AA F5 45 AA AA 15 55 AA AA 55
RG2 45 AA AA 55 7F AE 2B D7 55 AA EA 55 55 AA AB F5
RG3 54 AA AA 55 55 AF 9A 55 55 AA AA 55 D5 BB AA 55
RG4 15 AA A2 55 5D BA EE 57 55 88 AA 51 57 BA EA 5D
RG5 55 AA 8A 15 5D AA A9 55 55 8A AA 55 57 AA A2 5F
RG6 55 AA 0A 55 57 AB FA D5 51 2A AA 84 75 FA BA 55
RG7 55 AA AA 15 57 AB AA 5D 51 22 AA 15 75 AB AA 55
RG8 55 A8 AA 55 55 BB AA 55 41 A2 A2 40 5F BA BB 5D
RG9 55 AA AA 55 75 BA AA 75 55 AA AA 70 D5 AA AE D5
RGA 71 AA AE 55 D5 EA BA 57 55 2A A2 55 55 BE BA 55
RGB 55 AA AA D5 D5 AE AA 55 55 AA AA 55 D5 BE AA D5
RGC 51 AA A8 55 55 EA FA 5D 55 AA 8A 45 75 AA AA D5
RGD 55 8A AA 55 55 AA AA 55 57 AA 8A 55 55 AB EA D5
RGE 45 AA 8A 55 55 2A AA 5D 55 AA AA D5 7D FE AE 5D
RGF 55 AA AA 55 55 FB EB F5 D5 AA AA 75 57 FA AA 55
>
```

Example: Modify the contents of general purpose register "15".

```
>RG15<CR>
 RG15 AA 33<CR>
 RG16 AA<ESC>
>
```

Example: Display all special purpose register contents. Note that the symbol "--" indicates write only registers.

```
>RS
C0  C1  C2  C3  C4  C5  C6  C7  C8  C9  CA  CB  CC  CD  CE  CF
55  AA  A2  55  75  AA  AA  55  AA  FA  AA  55  77  AE  AB  75

SYSTEM REGISTERS:
FLAG RP0  RP1  SPH  SPL  IPH  IPL  IRQ  IMR  SYM
 D5   D6   D7   D8   D9   DA   DB   DC   DD   DE
 30   C0   C8   00   00   00   00   00   00   60

MODE AND CONTROL REGISTER:
BANK 0 REGISTERS
C0CT C1CT C0CH C0CL C1CH C1CL UTC  CRC  UIE  UIO  POM   PM
 E0   E1   E2   E3   E4   E5   EB   EC   ED   EF   F0   F1
 0A   08   00   00   00   00   02   00   00   00   00   --

H0C  H1C  P4D  P40D P2AM P2BM P2CM P2DM P2AI P2BI EMT  IPR
 F4   F5   F6   F7   F8   F9   FA   FB   FC   FD   FE   FF
 --   --   FF   00   --   --   --   --   00   00   7C   00

BANK 1 REGISTERS
COM C1M  COTH COTL C1TH C1TL DCH  DCL  UBGH UBGL UMA UMB WUMC WUMS
 E0  E1  E2   E3   E4   E5   F0   F1   F8   F9   FA   FB   FE   FF
 20  00  00   00   00   00   00   00   00   00   00   00   00   00
>
```

Example: Modify the contents of the system register or registers C0-CF (used as working registers only).

```
>RSC0<CR>
RSC0 55 11<CR>
RSC1 AA <ESC>
>
```

Example: Modify the contents of mode and control register in BANK1 at E2.

```
>RS1E2<CR>
RS1E2 00 FF<CR>
RS1E3 00 <ESC>
>
```

5.9 Jump/Branch Command - J

J address

J is the command for Jump/Branch.

address is a hexadecimal address of the emulation processor's program memory where the program counter is to be set.

Changing the emulation processor's program counter causes subsequent emulation to continue from that address in program memory space.

Example: Change the program counter from the current address to 5H.

```
>J6
>R
  A  B  C  D  E  H  L  I  F    X    Y    S   PC
 02 03 FE FD EF FB FF 00 FF FFFF FEFF FFFB 0006
>
```

CHAPTER 6

CONTROL SIGNAL COMMANDS

These commands enable or disable specific control signals going to the emulation processor. The available signal is either enabled or disabled at the input pin on the emulation processor. Only sense input to the processor is affected; circuits which generate the control signal in the target system are unaffected. The control characters and signal designation for hardware control signals (including pin numbers) and software control signals that are affected by the Disable and/or Enable commands are listed below:

Signal Designation	Z80	NSC800	Z8	Z8
B - Bus Hold Request	$\overline{BUSRQ}$ - 25 $\overline{BUSAK}$ - 23	$\overline{BREQ}$ - 36 $\overline{BACK}$ - 35		
I - Interrupt Request	$\overline{INT}$ - 16	$\overline{RSTA}$ - 22 $\overline{RSTB}$ - 23 $\overline{RSTC}$ - 24 $\overline{INTR}$ - 25	IRQ0 - 12 IRQ1 - 30 IRQ2 - 39 IRQ3 - 5	
N - Non-Maskable Interrupt Request	$\overline{NMI}$ - 17	$\overline{NMI}$ - 21		
R - System Reset	$\overline{RESET}$ - 26		$\overline{RESET}$ - 6	$\overline{RESET}$ - 26 (40-pin) 30 (48-pin)
X - Specify Program Memory and Data Memory Space			*	*

* 1. This is a software control function; no pin signals are affected. Emulation memory (HUEM/SUEM) is configured by using the following commands.

DX - combined memory space
EX - segregated memory space

2. A/Z/M/T/U and Download commands are not affected by EX or DX.

3. After power-up, or reset by the X command, MICE is set to DX mode (i.e. external memory space is combined).

Notes 1) NEW MICE-II NSC800 - to control emulation of the processor, MICE shares the use of WAIT input (pin-38), pulling the signal low to suspend emulation.

2) NEW MICE-II Z8 - D and E commands cannot control internal interrupt.

3) NEW MICE-II ZS8 - D and E commands cannot control internal or external interrupt.

6.1 Enable/Display Control Signal Command - E

E[control-signal]

E	is the command for Enable. If no signals are specified, keying in "E" will display the control signals that are enabled.
control-signal	is a single alphabetic character indicating a control signal to the emulation processor.

The control signal specified is activated at the pin of the emulation processor. When resetting the processor with the X command (section 5.7), these signals are not activated unless MICE is connected to a power-applied target system.

Example: Activate interrupt request control signal.

```
>E I
>
```

Example: Display the enabled control signals.

```
>E
ENABLE  -INT
>
```

6.2 Disable/Display Control Signal Command - D

D[control-signal]

D	is the command for Disable. If no signals are specified, keying in "D" will display the disabled control signals.
control-signal	is a single alphabetic character indicating a control signal to the emulation processor.

The control signal specified is deactivated at the pin of the emulation processor. When resetting the processor using the X command (section 5.7), these signals are activated if MICE is connected to a power-applied target system.

Example: Deactivate the interrupt request (INT) control signal, thereby not processing any interrupts which may occur. Note that this command does not affect the interrupt mask list in the processor status register.

```
>DI
>
```

Example: Display the disabled control signals.

```
>D
DISABLE  -NMI
         -INT
         -DMA
         -RESET
>
```

CHAPTER 7

EMULATION AND TRACE CONTROL COMMANDS

These commands perform free-running emulation or tracing for the emulation processor in real-time. Program operation may be recorded with user defined trigger addresses to specify the start or end of tracing. Up to 2048 machine cycles can be recorded.

Status information displayed by these commands use the following column headings: "**IFADDR ADDRESS DATA STATUS SPARE(8 BITS)**"

where: **IFADDR** is the instruction fetch address.
ADDRESS is the hexadecimal value on the address bus.
DATA is the hexadecimal value on the data bus.
STATUS is the type of processor activity.
SPARE is the status of the headers on the personality (CEP) board.

The specific types of processor activity that can be displayed in the status column are indicated below. The mnemonics listed for each processor also serve as qualifiers for H/F/B/L and BPP commands, as well as being displayed as statuses for C/SZ commands.

CPU Statuses	Z80	NSC	Z8	ZS8
Instruction Fetch Cycle	S	S	S	S
Memory Read Cycle	R	R	R	R
Memory Write Cycle	W	W	W	W
Port Input	I	I		
Port Output	O	O		
Interrupt Acknowledge	A	A		
Interrupt Cycle, Instruction Fetch			AS	
Interrupt Cycle, Memory Read			AR	AR
Interrupt Cycle, Memory Write			AW	AW

The following status designators apply only for the NEW MICE-II Z80 and NSC800.

1. The statuses "E" and "?" may be displayed for Cycle/ Instruction Step and List Trace commands; where "E" indicates a second opcode fetch cycle, and "?" indicates either a first or second opcode fetch cycle which NEW MICE-II cannot distinguish. Remember that these statuses cannot be used as qualifiers for command specification.

2. When selecting "S" activity as a qualifier in the List Trace command, the displayed messages will include "S", "E" and "?" qualifiers.

3. When using a qualifier in the Halt or Trace commands, if the breakpoint or trigger address is at the second opcode, then the qualifier "S" should be specified.

Trace point connections are also provided for monitoring external signals. There is a 8 post stick header on the CEP board designated X0-X7 from right to left. Any cable or wiring can be used to connect from these test points to the target. The monitored signals are called spare bits and support con-current trace for the address, data and status bus. Any of these bits can be monitored by trace commands to provide hardware status for display (under the SPARE column) in List, Cycle Step and Instruction Step commands; where "1" represents a high (or floating) signal and "0" indicates a low signal.

7.1 Go/Execution Command - G

G[address]

G is the command for Go/Execution.

address is a hexadecimal address of the emulation processor's program memory where emulation will begin.

The emulation processor's program counter is first set to the address indicated. NEW MICE-II then starts real-time emulation of the target processor. If no address is specified, program emulation will start at the current program counter.

Example: Display register content (PC=0006H), and start emulation from address 0H.

```
>R
  A  B  C  D  E  H  L  I  F    X    Y    S   PC
 02 03 FE FD EF FB FF 00 FF FFFF FEFF FFFB 0006
>GO
>
```

Example: Display register content, and resume emulation from the current PC.

```
>R
  A  B  C  D  E  H  L  I  F    X    Y    S   PC
 20 03 FE 10 5D FB FF 00 02 0099 FEFF 1FFA 0108
>G
>
```

Note: ?/!/D/E/L/BS* commands may be input without halting emulation, while the processor is executing G/FR/BR commands, or while executing F/B commands after the trigger has been matched. (* BS is a BPP command.)

7.2 Halt/Breakpoint Set Command - H

H[0[1|2]|1 [addx[count[qualifier]]]|2 [address-2]]

H is the command for Halt/Breakpoint Set.

0 is the command to clear a breakpoint.

1 is breakpoint 1.

2 is breakpoint 2. (Count, qualifier and wildcard do not apply.)

addx is an up to 4 digit hexadecimal address setting for breakpoint 1, with a wildcard byte pair of "XX" or "XXXX".

count is a hexadecimal value from 1 to 4000H which specifies the number of times the indicated condition must be matched before emulation stops.

qualifier is a single alphabetic character indicating the type of processor activity to be selected.

address-2 is an up to 4 digit hexadecimal address setting for breakpoint 2. (No wildcard, count or qualifier is permitted.)

7.2.1 Halt

H

Input "H" during emulation or tracing to stop execution immediately and display the current address.

Example: Stop emulation and display the current address.

```
>G
>H
      PROGRAM BROKE AT ADDRESS 0106
>
```

7.2.2 Set Breakpoint

H 1[addx[count [qualifier]]]|2 address-2

NEW MICE-II supports two real-time breakpoints. Breakpoint 1 can be used to specify a program address where emulation will stop. A count may also be specified to define the number of times the indicated condition is to be matched before emulation stops. If no count is specified, the default is one and emulation stops at the first breakpoint address matched. A qualifier can also be selected to choose the type of processor activity which must be matched along with the breakpoint address and count to stop emulation. If no qualifier is specified, all machine cycles are chosen. To enter a qualifier, a count must first be defined.

Breakpoint 2 can be used to specify a program address where emulation or tracing will stop. When setting breakpoint 2, set the breakpoint address only.

Example: Set breakpoint 1 to address 7H and count to 1.

```
>H1 7 1
>
```

Example: Set breakpoint 2 to address 1BH.

```
>H2 1B
>
```

Breakpoints 1 and 2 only set breakpoint conditions. Unless the emulation CPU is in execution, an emulation or trace command must be input to start the CPU. If H1 or H2 is matched, NEW MICE-II will stop the emulation processor and display the current address (at the specified breakpoint address). Breakpoint 1 will reset automatically after the Trace command since it shares the same logic as the Forward/Backward Trace commands. Note that breakpoints are not active in Cycle Step and Instruction Step commands.

Note: For NEW MICE-II Z8 and ZS8, when the breakpoint is matched, the CPU will finish executing the current instruction cycle and stop at the opcode fetch of the next instruction.

Example: Set an emulation program at 0H; then set breakpoint 1 to 3H, count to 1, qualifier for instruction fetch cycle (S), and breakpoint 2 to 1BH.

```
>J0
>H1 3 1 S
>H2 1B
>G
PROGRAM BROKE AT ADDRESS 0003
>
```

Note: Z80 (WAIT mode) will display: **"PROGRAM BROKE AT ADDRESS XXXX"**.

Example: Set the breakpoint address to XXFFH, the count to 1 and qualifier for memory write cycle (W).

```
>H1 XXFF 1 W
>G
PROGRAM BROKE AT ADDRESS 1FFE
>
```

7.2.3 Display Breakpoint

H 1|2

Input H1 or H2 to display breakpoint 1 or breakpoint 2 messages respectively.

Example: Display breakpoint 1 and 2 messages. Note that the address, count and qualifier are displayed for breakpoint 1; and only address is displayed for breakpoint 2.

```
>H1
H1= XXFF 1 W
>H2
H2= 1B
>
```

7.2.4 Clear Breakpoint

H O[1|2]

NEW MICE-II can clear all breakpoints concurrently or separately.

Example: Clear breakpoint 1 only.

```
>HO1
>H1
H1=
>
```

Example: Clear all breakpoints concurrently.

```
>HO
>H1
H1=
>H2
H2=
>
```

7.3 Forward Trace Command - F

F[R]addx[count[qualifier]]

F is the command for Forward Trace.

R continues running the emulation processor after the trace stops.

addx is an up to 4 digit hexadecimal trigger address with a wildcard byte pair of "XX" or "XXXX" where NEW MICE-II begins recording trace information.

count is a hexadecimal value from 1 to 4000H which specifies the number of times the target condition must be matched before the trace records information.

qualifier is a single alphabetic character indicating the type of processor activity that must be matched with the trigger address.

Forward tracing starts real-time emulation of the target and begins recording target status information when the trigger condition is matched. Forward trace will stop when the trace buffer is full or breakpoint 2 is reached. Forward trace can record up to 2048 machine cycles.

When the trace stops, MICE halts the emulation CPU at the break address. The recorded information can be examined by using the List Trace Buffer command.

Note: For NEW MICE-II Z8 and ZS8, when the breakpoint is matched or the trace buffer filled, the CPU will finish executing the current instruction cycle and stop at the opcode fetch of the next instruction.

Option R allows the emulation CPU to continue running after the trace stops but breakpoint 2 will force the CPU to stop. Without option R, the CPU will stop whenever the trace stops. If only the trigger address is specified, all machine cycles after that trigger address is encountered are recorded. If a count is specified, the trace will record information after the number of matches of the indicated trigger condition occur. If no count is specified, the trace will begin recording information as soon as the trigger condition is matched.

A single qualifier may be specified to choose the type of processor activity associated with the trigger address; default is for all machine cycles. To enter a qualifier, a count must first be defined. When specified, the trace will start recording only when the trigger address together with the qualifier occurs the number of times defined by count.

Example: Begin recording program status from the trigger address of 3H. In the following message, **STEP** indicates the last trace frame recorded. The trace count begins at zero, so the actual value is one more than the step number shown. The trace buffer is completely filled if STEP 07FF is displayed.

```
>F3
THE TRACE STOPS AT STEP 07FF
>
```

Depending on command specifications, the elapse time before the buffers fill may be long. Entering <ESC> terminates the trace, yet retains the information already recorded.

Example: Specify a trigger address of 112H which the program never reaches; the trace buffer is emptied. Note that if breakpoint 2 is encountered before the trigger address is reached, the message will be displayed without entering <ESC>.

```
>F112
 <ESC>
 FORWARD TRACE FAILS
>
```

Example: Begin tracing when the memory location 2H is addressed and stop at breakpoint 2 before the trace buffer is filled.

```
>J0
>H2 7
>F2
THE TRACE STOPS AT STEP 00BD
PROGRAM BROKE AT ADDRESS 0007
>
```

Example: Begin recording status information the first time the program writes to an address in the range of 1F00H to 1FFFH. Continue running after the trace stops until the program addresses 001BH. Note that qualifiers are only associated with the trigger address; all machine cycles are recorded after the trace has begun.

```
>J0
>H2 1B
>FR 1FXX 1 W
THE TRACE STOPS AT STEP 01A1
PROGRAM BROKE AT ADDRESS 001B
>
```

7.4 Backward Trace Command - B

B[R]addx[count[qualifier]]

B	is the command for Backward Trace.
R	continues running the emulation processor after the trace stops.
addx	is an up to 4 digit hexadecimal trigger address with a wildcard byte pair of "XX" or "XXXX" where emulation is to stop.
count	is a hexadecimal value from 1 to 4000H which specifies the number of times the target condition must be matched before the trace stops recording information.
qualifier	is a single alphabetic character indicating the type of processor activity that must be matched with the trigger address.

Backward tracing starts real-time emulation of the target and immediately begins recording target status information until the trigger address or breakpoint 2 is reached. When the trace is matched, NEW MICE-II halts the emulation processor at the specified break address.

Note: For NEW MICE-II Z8 and ZS8, when the breakpoint is matched or the trace buffer filled, the CPU will finish executing the current instruction cycle and stop at the opcode fetch of the next instruction.

Backward trace can record up to 2048 cycles and the recorded information can be examined using the List Trace Buffer command. If more than 2048 cycles elapse before backward tracing ends, only the last 2048 cycles are recorded. Option R allows the emulation processor to continue running after the trace stops, however breakpoint 2 will force the CPU to stop and display a breakpoint match message.

Without option R, the CPU will stop whenever the trace stops. If only the trigger-address is specified, all machine cycles are recorded up to and including the trigger-address encountered. If a count is specified, backward tracing will record all information before the number of matches of the indicated trigger-address. If no count is specified, the trace will stop after the first match.

A single qualifier may be specified to choose the type of processor activity associated with the trigger-address. To enter any qualifier, a count must first be defined.

Example: Begin tracing immediately and stop when the memory location 1BH is addressed. Note that the trace will also stop when breakpoint 2 is encountered.

```
>J0
>H2 7
>B1B
 THE TRACE STOPS AT STEP 00BF
 PROGRAM BROKE AT ADDRESS 0007
>
```

In the above example, the breakpoint message following the command line indicates that 07H cycles have been recorded. The counter is a hexadecimal value from 0 to 7FFH. The trace count begins at zero, so the actual value is one more than the step number shown.

Again, depending on the specification, elapse time before the trigger address is encountered may be long. Entering an <ESC> terminates the trace, yet retains the information already recorded.

Example: Specify a trigger address of 112H which the program never reaches. The trace buffer is filled with the cycles immediately before <ESC> is received.

```
>H0
>B112
<ESC>
THE TRACE STOPS AT STEP 07FF
>
```

Example: Begin tracing immediately and stop the 1st time the program reads from the address range of 1F00H to 1FFFH. Continue running after the trace stops, until the program addresses 001BH. Note that qualifiers are only associated with the trigger address; all machine cycles are recorded once the trace begins.

```
>J0
>H2 1B
>BR 1FXX 2 W
THE TRACE STOPS AT STEP 0007
PROGRAM BROKE AT ADDRESS 001B
>
```

7.5 List/Display Trace Buffer Command - L

L[step[address-1[address-2[qualifier(s)]]]|S[step]|Z[step]|N]

L	is the command for List/Display Trace Buffer.
step	is a hexadecimal value from 0 to 7FFH indicating the initial cycle step to display.
address-1	is the starting address of the range to be listed.
address-2	is the ending address of the range to be listed.
qualifier(s)	are single characters indicating the type of processor activities to list.
S	lists the trace buffer using mnemonic code.
Z	lists the trace buffer displaying mnemonic code and all machine statuses.
N	displays the frame number where the last trace stopped.

Status information recorded during a trace command is displayed using this command. If a step is specified, NEW MICE-II lists the trace buffer from the specified step to the last recorded step. If no step is specified, NEW MICE-II will list all records in the trace buffer. An optional address range can be entered to specify the

range to be listed. If no address range is specified, then the display range is from 0000H to FFFFH.

The type of information to be listed is specified by entering qualifier(s). If no qualifier is entered, all machine cycles within the specified address range are listed. To enter a qualifier, an address range must first be defined. To enter an address range, a step must first be entered. The List command accepts multiple qualifiers; and displays only machine cycles that match the specified qualifiers. If the buffer is listed with the LS or LZ command, only "step" can be specified.

Example: List all records in the trace buffer. Note that the list operation can be terminated by entering an <ESC>. Note that **FRAME** is the sequence number of the trace step in hexadecimal; the range is 0 to 2047 (7FFH).

```
>L
 FRAME IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
  0000         0000   31   ?      11111111
  0001         0001   00   R      11111111
  0002         0002   20   R      11111111
  0003   0003  0003   CD   S      11111111
  0004         0004   00   R      11111111
  0005         0005   01   R      11111111
  0006         1FFF   00   W      11111111
  0007         1FFE   06   W      11111111
>
```

Example: List the trace buffer from the 3th step to the last step, and terminate the operation before the listing is completed.

```
>L2
 FRAME IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
  0002         0002   20   R      11111111
  0003   0003  0003   CD   S      11111111
  0004         0004   00   R      11111111
  0005         0005   01   R      11111111
  0006         1FFF   00   W      11111111
  0007         1FFE   06   W      11111111
>
```

Example: List the trace buffer write cycles in the range 3H to 5H starting at step 2.

```
>L2 3 5
 FRAME IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
  0003         0003   CD   S      11111111
  0004         0004   00   R      11111111
  0005         0005   01   R      11111111
>
```

Example: List the trace buffer from 0H to FH, with qualifier S.

```
>L0 0 F S
 FRAME IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
  0000         0000   31   ?      11111111
  0003   0003  0003   CD   S      11111111
>
```

Example: List trace message with source code; the operation is terminated before the listing is completed. Note that the command also ends if the trace buffer is empty or if nothing is recorded within the specified range.

```
>LS
 FRAME IFADDR ADDRESS DATA          SOURCE CODE
  0000    0000    0000   31          LD    SP,2000
  0003    0003    0003   CD          CALL  0100
>
```

Example: List trace message with mnemonic code and all machine statuses; then display the frame number of the last trace.

```
>LZ
 FRAME IFADDR ADDRESS DATA STATUS    SPARE       SOURCE CODE
  0000          0000   31    ?      11111111
  0001          0001   00    R      11111111
  0002          0002   20    R      11111111
  0003   0003   0003   CD    S      11111111     CALL  0100
  0004          0004   00    R      11111111
  0005          0005   01    R      11111111
  0006          1FFF   00    W      11111111
  0007          1FFE   06    W      11111111
>
>LN
THE TRACE STOPS AT STEP 0007
>
```

CHAPTER 8

STEPPED EMULATION COMMANDS

These commands perform cycle-stepped or instruction-stepped emulation of the emulation processor in real time. Note that breakpoints are not active for C/S commands. Refer to page 7-1 for a description of the displayed status information.

Note: A blank space (ASCII 20H) is sent at the end of each displayed line immediately before the cursor for C/S commands. This code serves as an end of data message to the host computer; and is provided to help in implementing symbolic debuggers.

8.1 Cycle Step Command - C

C[count]

C is the command for Cycle Step.

count specifies the machine cycle interval between displayed messages. The range is 0-FFFFH. Note that an input value of "0" represents 10000H.

The Cycle Step command first stops the emulation processor, then steps and executes one program one cycle, stopping the processor in HALT state (**where the displayed target status has already been executed**). NEW MICE-II then awaits a <CR> or <LF> to execute and display the next machine cycle. Enter a <CR> or <LF> to repeat the entire sequence, or an <ESC> to terminate the single-cycle mode of emulation.

Because the Z8 and ZS8 cannot be stopped, MICE keeps the CPU running in place with a jump instruction, **where the displayed instruction (including all associated machine cycles) has already been executed.**

However, note that the Z80 (WAIT Mode) stops the emulation processor at WAIT state instead, **where the displayed cycle status has not yet been executed.** Because the displayed status is still active in this situation, debug for associated hardware signals is simplified.

Example: Cycle step the following program.

```
>JO
>C
      IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
        0000    0000   31     S       11111111 <CR>
                0001   00     R       11111111 <CR>
                0002   20     R       11111111 <CR>
        0003    0003   CD     S       11111111 <CR>
                0004   00     R       11111111 <CR>
                0005   01     R       11111111 <CR>
                1FFF   00     W       11111111 <CR>
                1FFE   06     W       11111111 <CR>
        0100    0100   11     S       11111111 <CR>
                0101   00     R       11111111 <CR>
                0102   10     R       11111111 <CR>
        0103    0103   3E     S       11111111 <ESC>
>
```

Example: Cycle step the above program using a count of 3.

```
>JO
>C3
      IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
                0002   20     R       11111111 <CR>
                0005   01     R       11111111 <CR>
        0100    0100   11     S       11111111 <CR>
        0103    0103   3E     S       11111111 <CR>
                1000   20     W       11111111 <CR>
        0108    0108   FE     S       11111111 <CR>
                010B   03     R       11111111 <ESC>
>
```

8.2 Instruction Step Command - S

S[S|R][count]|Z

S — is the command for Instruction Step.

S(option) — displays messages in mnemonic form.

R — displays register contents in addition to mnemonic code.

count — is an hexadecimal value from 0000H to FFFFH specifying the step interval between status display. (Note that 0000H indicates 10000H steps.)

Z — displays cycle status in addition to mnemonic code.

8.2.1 Instruction Step

S

If count is not specified for instruction step, the default is one. In this mode, MICE first **displays the current instruction to be executed** and waits for input. To cause the emulation processor to execute this instruction, a <CR> or <LF> should be entered; the new status is then displayed and MICE again waits. Enter a <CR> or <LF> to repeat the entire sequence, displaying the new status after every instruction. Enter <ESC> to terminate the single step mode of emulation. (Remember that breakpoints are not active for the S command.)

Instruction step only displays "S" status to indicate an instruction fetch cycle; no other status types are displayed. If a more careful examination of a program is required, first Instruction Step to the problem area, and then Cycle Step through the problem area for a more detailed display.

Example: Instruction step the previous program. Note that the processor does not begin emulation until the current status displays and either a <CR> or <LF> is entered.

```
>J0
>S
       IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
         0000    0000   31    S      11111111 <CR>
         0003    0003   CD    S      11111111 <CR>
         0100    0100   11    S      11111111 <ESC>
>
```

8.2.2 Instruction Step in Mnemonic Form

SS

Example: Instruction step with S option.

```
>J0
>SS
        IFADDR ADDRESS DATA        SOURCE CODE
          0000    0000   31        LD     SP,2000 <CR>
          0003    0003   CD        CALL   0100 <CR>
          0100    0100   11        LD     DE,1000 <ESC>
>
```

8.2.3 Instruction Step with Register Content

SR

The R option displays register content along with mnemonic information. All register contents associated with the current step in program execution can be observed; but remember that the contents displayed are prior to executing the instruction.

Example: Instruction step with R option.

```
>J0
>SR
       IFADDR ADDRESS DATA        SOURCE CODE
 A=00 B=00 C=06 D=10 E=01 H=F7 L=FF I=00 F=9B X=300F Y=FFFF S=1FFE PC=0000
         0000   0000   31         LD     SP,2000 <CR>
 A=00 B=00 C=06 D=10 E=01 H=F7 L=FF I=00 F=9B X=300F Y=FFFF S=2000 PC=0003
         0003   0003   CD         CALL   0100 <ESC>
>
```

The SR command for NEW MICE-II Z8 and ZS8 displays working register content for the current instruction step in addition to mnemonic code (refer to RW command, section 5.8)

Example: Execute instruction-step with mnemonic code for NEW MICE-II Z8.

```
>SR
   IFADDR ADDRESS DATA     SOURCE CODE
    000C     000C    31     SRP   #40
RP    0   1   2   3   4   5   6   7   8   9   A   B   C   D   E   F
40   00  00  00  00  47  00  00  00  00  00  00  10  00  00  00  00<CR>
    000E     000E    4C     LD    R04,#AA
RP    0   1   2   3   4   5   6   7   8   9   A   B   C   D   E   F
40   00  00  00  00  AA  00  00  00  00  00  00  10  00  00  00  00<CR>
    0010     0010    31     SRP   #50
RP    0   1   2   3   4   5   6   7   8   9   A   B   C   D   E   F
50   00  00  00  00  AA  00  00  00  00  00  00  10  00  00  00  00<CR>
    0012     0012    4C     LD    R04,#55
RP    0   1   2   3   4   5   6   7   8   9   A   B   C   D   E   F
50   00  00  00  00  55  00  00  00  00  00  00  10  00  00  00  00<ESC>
>
```

Example: Execute instruction-step with mnemonic code for NEW MICE-II ZS8.

```
>SR
   IFADDR ADDRESS DATA     SOURCE CODE
    0020    0020    FF     NOP
RP0(C0) R0=11 R1=AA R2=A2 R3=55 R4=75 R5=AA R6=AA R7=55  CZSVDHFB  PC=0021
RP1(C8) R8=AA R9=FA RA=AA RB=55 RC=77 RD=AE RE=AB RF=75  00110000<CR>
            0021    31     SRP    #50
RP0(50) R0=55 R1=AA R2=8A R3=15 R4=5D R5=AA R6=A9 R7=55  CZSVDHFB  PC=0023
RP1(58) R8=55 R9=8A RA=AA RB=55 RC=57 RD=AA RE=A2 RF=5F  00110000<CR>
            0023    31     SRP    #E0
RP0(E0) R0=0A R1=08 R2=00 R3=00 R4=00 R5=00 R6=-- R7=--  CZSVDHFB  PC=0025
RP1(E8) R8=-- R9=-- RA=-- RB=02 RC=00 RD=00 RE=-- RF=00  01010000<CR>
            0025    FF     NOP
RP0(E0) R0=0A R1=08 R2=00 R3=00 R4=00 R5=00 R6=-- R7=--  CZSVDHFB  PC=0026
RP1(E8) R8=-- R9=-- RA=-- RB=02 RC=00 RD=00 RE=-- RF=00  01010000<ESC>
>
```

8.2.4 Instruction Step with Count

S[S|R] count

If an instruction count of one (default) is entered, the current instruction to be executed is displayed; NEW MICE-II does not step. If an instruction count other than one is entered, emulation immediately begins from the current program counter and continues until the count of the next instruction to be executed equals the number specified. The current status is then displayed and NEW MICE-II waits for a <CR> or <LF> before executing the next "count" instruction. Enter an <ESC> to terminate the multi-step mode of emulation. For large intervals, emulation may be terminated by entering an <ESC> before the specified number of instructions are executed.

Instruction Step executes "count-1" instructions before the first status line is displayed and then executes "count" instructions between subsequent displays.

Example: Multi-step the program and compare results with the first example.

```
>J0
>S3
      IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
        0007   0007   CD     S      11111111 <CR>
        0103   0103   3E     S      11111111 <CR>
        0107   0107   E5     S      11111111 <CR>
        010D   010D   E1     S      11111111 <ESC>
>
```

8.2.5 Instruction Step with Cycle Status

SZ

The SZ option includes a memory and I/O access message that provides data for bus cycles R/W/I/O/A [e.g. 7FEF-W-12]. The first four digits (7FEF) indicate the memory address; the capital letter (W) represents the type of bus cycle; and the last two digits (12) are the data transferred.

Example: Instruction step the program displaying Cycle status together with mnemonic code.

```
>JO
>SZ
       IFADDR ADDRESS DATA          SOURCE CODE
         0000   0000   31          LD    SP,2000 <CR>
         0003   0003   CD          CALL  0100 <CR>
                1FFF   -W-  00
                1FFE   -W-  06 <CR>
         0100   0100   11          LD    DE,1000 <CR>
         0103   0103   3E          LD    A,20 <ESC>
>
```

8.3 Application Notes for Stepped Emulation Commands

1. To make debugging easier, there are two available methods for slowing down the CPU's execution speed.

   a) Use the C command with a "Ø" count to slow execution speed as follows:

NEW MICE-II	Speed
Z80	1/150
NSC800	1/130
Z8	1/708
ZS8	1/454

   b) Use the S command with a "Ø" count to slow execution speed as follows:

NEW MICE-II	Speed
Z80	1/7980
NSC800	1/6450
Z8	1/710
ZS8	1/459

   c) Typing ahead with <CR> will keep the processor running.

   Example: >CØ
   <CR>
   <CR>
   <CR>

CHAPTER 9

UTILITY COMMANDS INVOLVING A SYSTEM

These commands are only applicable when NEW MICE-II is connected to a host system. Programs and data can be downloaded from a file in a host computer to memory in the target system. Conversely, information can also be uploaded from the target back to the host.

Intel and Tektronix loading formats, both of which are recognized by MICE, are described in detail in the following pages.

Note: For the microcontrollers Z8 and ZS8, Upload and Download commands can only access emulation program memory, i.e. memory type "P". (Refer to section 2.6 for a detailed description of memory types).

9.1 Download Command (Intel Format) - :

:load-record

: is the command keyword for Download.

load-record is a string of ASCII data containing up to 256 bytes of program information.

Each record transferred contains the record type, length, memory load address, and checksum in addition to data. Each transfer is limited to 256 bytes of program data. The general format of a record, shown with spaces separating each field, is:

RECORD MARK	RECORD LENGTH	LOAD ADDRESS	RECORD TYPE	PROGRAM DATA	CHECK-SUM
:	##	**aaaa**	**tt**	**dd...dd**	**cc**

where:

: is the keyword used to signal start of record.

is a two ASCII hexadecimal value indicating the record length, the number of data bytes in the record.

aaaa is a four ASCII hexadecimal value indicating the program memory load-address, the address at which the first byte is to be loaded. (For record type 01 [next item], this field contains "0000".) Successive data bytes are stored in the following memory locations.

tt is a two ASCII hexadecimal value representing the record type:

tt	##
00 - data record	actual data length
01 - end of file record	00

dd...dd is a two ASCII hexadecimal value per byte representation of the program.

tt	dd...dd
00	A pair of hex digits representing the ASCII code for each data byte, where the high order digit is the 1st digit of each pair.
01	none

cc is a two ASCII hexadecimal value representing the negative sum of the record. Beginning with the record length "##" and ending with the checksum "cc", the hexadecimal sum, taken two at a time, modulo 256 should be zero.

When NEW MICE-II receives a ":", it reads the record length (##). The rest of the record is then read and verified. If the incoming checksum agrees with the computed checksum, NEW MICE-II stores the data into program memory of the target system and reprompts after the following acknowledgement response <ACK sequence> is sent:

ACK LF CR NUL NUL NUL NUL NUL NUL

If the checksum does not agree, NEW MICE-II also reprompts but the alternate negative acknowledgement response <NAK sequence> is sent:

NAK LF CR NUL NUL NUL NUL NUL NUL

Example: Download with an end-record into a target system using Intel format. Note that the received characters are not echoed. A <CR> is not required at the end of the input line.

```
>:060000002300A8A917046B<ACK sequence>
>:00000001FF<ACK sequence>
>
```

In the above example, the first load record is

```
##      = 06H
aaaa    = 0000H
tt      = 00H
dd...dd = 23H, 00H, A8H, A9H, 17H, 04H
cc      = 6BH
```

where: 06H+00H+00H+00H+23H+00H+A8H+A9H+17H+04H+6BH = 00H
For the end-record,

```
##      = 00H
aaaa    = 0000H
tt      = 01H
cc      = FFH
```

where: 00H+00H+00H+01H+FFH = 00H

Example: Download with an end-record into a target system using Intel type 2 format.

```
>:10000000004992DB246DB6FF4891DA263CB5FE47E5<ACK sequence>
>:0600100090D9226BB4FD43<ACK sequence>
>:020000020036C6<ACK sequence>
>:100000000062C42688EA4CAE1072D43698FA5CBE00<ACK sequence>
>:080010002082E446A80A6CCE30<ACK sequence>
>:00000001FF<ACK sequence>
>
```

End-of-record must then be specified by 01 in the last line of transmission.

9.2 Download Command (Tektronix Format) - /

/load-record

/ is the command keyword for Download.

load-record is a string of ASCII data containing up to 128 bytes of program information.

Each record transferred contains the record type, length, memory load address and checksum in addition to data. Each transfer is limited to 128 bytes of program data. The general format of a record, shown with spaces separating each field, is:

RECORD MARK	LOAD ADDRESS	RECORD LENGTH	LOAD SUM	PROGRAM DATA	CHECK-SUM
/	**aaaa**	**##**	**ss**	**dd...dd**	**cc**

where:

/ is the keyword used to signal start of record.

aaaa is a four ASCII hexadecimal value indicating the program memory load-address, the address at which the first byte is to be loaded. Successive data bytes are stored in the following memory locations.

is a two ASCII hexadecimal value indicating record length, the number of data bytes in the record. A record length of zero indicates end-of-file.

ss is a two ASCII hexadecimal value representing the sum of the preceding six digits, load-address and record length.

dd...dd is a two ASCII hexadecimal value per byte representation of the program.

cc is a two ASCII hexadecimal value representing the sum of the digits comprising the data, modulo 256.

When NEW MICE-II receives a "/", input is read until a <CR> terminates the line. Checksums are then computed and compared. If the incoming checksum agrees with the computed checksum, NEW MICE-II stores the data into program memory of the target system and reprompts after the following acknowledgement response <ACK sequence> is sent:

ACK LF CR NUL NUL NUL NUL NUL NUL

If the checksum does not agree, NEW MICE-II also reprompts but the alternate negative acknowledgement response <NAK sequence> is sent:

NAK LF CR NUL NUL NUL NUL NUL NUL

Example: Download with an end-record into a target system using Tektronix format.

```
>/000006062300A8A9170436<CR>
<ACK sequence>
>/00000000<CR>
<ACK sequence>
>
```

In the above example, the load record is

```
aaaa     = 0000H
##       = 06H
ss       = 06H
dd...dd  = 23H, 00H, A8H, A9H, 17H, 04H
cc       = 36H
```

where: ss = 0H+0H+0H+0H+0H+6H = 06H
cc = 2H+3H+0H+0H+AH+8H+AH+9H+1H+7H+0H+4H = 36H

For the end-record,

```
aaaa     = 0000H
##       = 00H
ss       = 00H
```

where: ss = 0H+0H+0H+0H+0H+0H = 00H

9.3 Upload Command - U

U start-address end-address [format]

U	is the command for Upload.
start-address	is a logical hexadecimal address of the emulation processor's memory where data transfer begins.
end-address	is the last logical hexadecimal address of the emulation processor's program memory where data transfer ends.
format	is a single alphabetic character [I|T] indicating the load-record format to be used.

I - Intel
T - Tektronix

Memory contents defined by the memory range start-address to end-address are transferred to the host system using either Intel or Tektronix format. If the format is not specified, then Intel is used. Also, the end-address must be greater than or equal to the start-address or an error message is sent and the command ended.

NEW MICE-II first sends the current segment value record to the host system. It then reads data from memory and sends a maximum of 32 bytes (depending on data length) to the host system using one of the above formats. Data transmission continues in this manner until an end address is sent. NEW MICE-II then sends an end-

of-record file and prompts for the next command.

If anything other than an <ACK> is received when NEW MICE-II is waiting for an acknowledgement response, the same block is retransmitted. If after five retries transmission is still unsuccessful, the command is aborted, and an error message is sent to the host system before NEW MICE-II prompts for the next command. The command can also be aborted by the host system when NEW MICE-II receives an <ESC> character from the console.

Example: Upload from NEW MICE-II using Intel format. Note that load records are shown on separate lines for clarity. No <CR> or <LF> characters are generated by NEW MICE-II, and anything received other than an <ACK>, is treated the same as a <NAK>.

```
>U 0 6 I<CR>
:070000002300A8917046E8FF<ACK>
:00000001FF<ACK>
>
```

Example: Upload using Tektronix format.

```
>U 0 6 T<CR>
/000007072300A8917046E848<CR>
<NAK>
/000007072300A8917046E848<CR>
<ACK>
/00000000
<ACK>
>
```

In the above example, the host system did not acknowledge the first transmission when Tektronix format was used. Hence, the first line was retransmitted until an <ACK> was received.

Appendix A

Summary of MICE-II Commands

MICE-II UTILITY COMMANDS

? - Help Command
! - Attention Command

MEMORY, PORT AND REGISTER COMMANDS

M - Memory Display/Examine/Modify/Fill/Search Command
M[start-address[end-address[data-1[...data-8][S]]]]

T - Memory Checksum/Test/Transfer/Compare Command
T start-address end-address {S|M|address-3[V]}

A - Line Assembly command
A[start-address]

Z - Disassembly Command
Z[start-address[end-address]]

I - Port Input Command
I port[count[duration]]

O - Port Output Command
O port data-1 [...data-8]

X - Reset/Initialization Command
X[address]

R - Register Display/Modify Command
R[register]

J - Jump/Branch Command
J address

CONTROL SIGNAL COMMANDS

D - Disable/Display Control Signal Command
D[control signal]

E - Enable/Display Control Signal Command
E[control signal]

EMULATION AND TRACE CONTROL COMMANDS

G - Go/Execution Command
G[address]

H - Halt/Breakpoint Set Command
H[0[1|2]|1[address-1[count[qualifier]]]|2 [address-2]]

F - Forward Trace Command
F[R]trigger-address[count[qualifier]]

B - Backward Trace Command
B[R]trigger-address[count[qualifier]]

L - List/Display Trace Buffer Command
L[step[address-1[address-2[qualifier(s)]]]|S[step]|Z[step]|N]

STEPPED EMULATION COMMANDS

C - Single Cycle/Step Command
C[count]

S - Instruction Step Command
S[[S|R][count]|Z]

UTILITY COMMANDS INVOLVING A SYSTEM

: - Download Command (Intel Format)
:load-record

/ - Download Command (Tektronix Format)
/load-record

U - Upload Command
Ustart-address end-address [I|T]

Appendix B

Hexadecimal-Decimal Conversion

To find the decimal equivalent of a hexadecimal number, first locate your hex number in the correct position (1st through 4th place digit) in the following table. Note the decimal equivalent for each hex digit in your number and then add up these decimal values.

To find the hexadecimal equivalent of a decimal number, locate the next lower decimal number in the table, write down the hex equivalent and its position (1st through 4th place digit). Subtract this decimal number from yours, take the difference and continue this process until conversion is completed.

BYTE				BYTE			
4th place digit		**3rd place digit**		**2nd place digit**		**1st place digit**	
HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC
0	0	0	0	0	0	0	0
1	4096	1	256	1	16	1	1
2	8192	2	512	2	32	2	2
3	12288	3	768	3	48	3	3
4	16384	4	1024	4	64	4	4
5	20480	5	1280	5	80	5	5
6	24576	6	1536	6	96	6	6
7	28672	7	1792	7	112	7	7
8	32768	8	2048	8	128	8	8
9	36864	9	2304	9	144	9	9
A	40960	A	2560	A	160	A	10
B	45056	B	2816	B	176	B	11
C	49152	C	3072	C	192	C	12
D	53248	D	3328	D	208	D	13
E	57344	E	3584	E	224	E	14
F	61444	F	3840	F	240	F	15

Appendix C

ASCII Code Lists and Definitions

Table C-1
ASCII Code List

HEX	DEC	CHAR	HEX	DEC	CHAR
00	0	NUL	18	24	CAN
01	1	SOH	19	25	EM
02	2	STX	1A	26	SUB
03	3	ETX	1B	27	ESC
04	4	EOT	1C	28	FS
05	5	ENQ	1D	29	GS
06	6	ACK	1E	30	RS
07	7	BEL	1F	31	US
08	8	BS	20	32	SP
09	9	HT	21	33	!
0A	10	LF	22	34	"
0B	11	VT	23	35	#
0C	12	FF	24	36	$
0D	13	CR	25	37	%
0E	14	SO	26	38	&
0F	15	SI	27	39	'
10	16	DLE	28	40	(
11	17	DC1	29	41	)
12	18	DC2	2A	42	*
13	19	DC3	2B	43	+
14	20	DC4	2C	44	,
15	21	NAK	2D	45	-
16	22	SYN	2E	46	.
17	23	ETB	2F	47	/

Appendix C

ASCII Code Lists and Definitions

Table C-1
ASCII Code List (Cont.)

HEX	DEC	CHAR	HEX	DEC	CHAR
30	48	0	46	70	F
31	49	1	47	71	G
32	50	2	48	72	H
33	51	3	49	73	I
34	52	4	4A	74	J
35	53	5	4B	75	K
36	54	6	4C	76	L
37	55	7	4D	77	M
38	56	8	4E	78	N
39	57	9	4F	79	O
3A	58	:	50	80	P
3B	59	;	51	81	Q
3C	60	<	52	82	R
3D	61	=	53	83	S
3E	62	>	54	84	T
3F	63	?	55	85	U
40	64	@	56	86	V
41	65	A	57	87	W
42	66	B	58	88	X
43	67	C	59	89	Y
44	68	D	5A	90	Z
45	69	E	5B	91	[

Appendix C

ASCII Code Lists and Definitions

Table C-1
ASCII Code List (Cont.)

HEX	DEC	CHAR	HEX	DEC	CHAR
5C	92	\	6E	110	n
5D	93	]	6F	111	o
5E	94	^	70	112	p
5F	95	-	71	113	q
60	96	`	72	114	r
61	97	a	73	115	s
62	98	b	74	116	t
63	99	c	75	117	u
64	100	d	76	118	v
65	101	e	77	119	w
66	102	f	78	120	x
67	103	g	79	121	y
68	104	h	7A	122	z
69	105	i	7B	123	{
6A	106	j	7C	124	|
6B	107	k	7D	125	}
6C	108	l	7E	126	~
6D	109	m	7F	127	DEL

Appendix C

ASCII Code Lists and Definitions

Table C-2
ASCII-Code Definitions

ABB	CTRL	MEANING	HEX
NUL		NULL Character	00
SOH	A	Start of Heading	01
STX	B	Start of Text	02
ETX	C	End of Text	03
EOT	D	End of Transmission	04
ENQ	E	Enquiry	05
ACK	F	Acknowledge	06
BEL	G	Bell	07
BS	H	Backspace	08
HT	I	Horizontal Tabulation	09
LF	J	Line Feed	0A
VT	K	Vertical Tabulation	0B
FF	L	Form Feed	0C
CR	M	Carriage Return	0D
SO	N	Shift Out	0E
SI	O	Shift In	0F
DLE	P	Data Link Escape	10
DC1	Q	Device Control 1	11
DC2	R	Device Control 2	12
DC3	S	Device Control 3	13
DC4	T	Device Control 4	14
NAK	U	Negative Acknowledgement	15
SYN	V	Synchronous Idle	16

Table C-2
ASCII-Code Definitions (Cont.)

ABB	CTRL	MEANING	HEX
ETB	W	End of Transmission Block	17
CAN	X	Cancel	18
EM	Y	End of Medium	19
SUB	Z	Substitute	1A
ESC		Escape	1B
FS		File Separator	1C
GS		Group Separator	1D
RS		Record Separator	1E
US		Unit Separator	1F
SP		Space	20
DEL		Delete	7F

Appendix D

Writing a NEW MICE-II Driver Program

If for some reasons user wishes to write his own NEW MICE-II driver program, the following guides are provided and should be considered in writing such program.

D.1 System Setup

User's system must have an RS-232C series communication port.

Communication protocol refer to section 2.5.

D.2 MICE Command Transparency

STEP	DRIVER ACTION	REMARKS
1	Sends 'CR' to try to initiate communication with MICE. If it receives no response, send the character which has just been input from keyboard. Receive all responses from MICE and display them until a prompt (>) followed by handshaking code appear on the screen.	;MICE responds with self-diagnostic results.
2	Pick up one key (keyboard input) say "char-X", and send it to MICE. Receive response from MICE and display it until MICE echo with message- "char-Y" where: If "char-X" is BS, receive two data If the second data is SP, "char-Y" is BS. else "char-Y" is handshaking code. If "char-X" is ESC, "char-Y" is handshaking code, else "char-Y" is the same as "char-X".	;BS (Back Space=08H) ;SP (Space bar=20H) ;ESC=1BH
3	If "char-X" is 'CR', Receive response from MICE and display received messages until a prompt (>) followed by handshaking code appear on the screen. At the same time, scan keyboard input during display and send any received data to MICE.	;CR=0DH
4	Goto Step 2.	

D.3 Upload

STEP	DRIVER ACTION	REMARKS
1	Send attention command to MICE. If displayed title includes version description, set flag.	;Command syntax: "! CR"
2	Send upload command to MICE. Receive echo messages until 'CR' is detected.	;Command syntax: "U start_adr end_adr CR"
3	If flag on, receive upload data until handshaking code is detected. else receive upload data counted by record length.	
4	Store record to file.	
5	Send ACK (CTRL-F) to MICE.	
6	If not end-record, goto Step 3. Otherwise close the file.	

D.4 Download

STEP	DRIVER ACTION	REMARKS
1	Get one record from file.	
2	Send one record to MICE and receive one character (e.g. char-Z) from MICE.	
3	If "char-Z" is ACK (Ctrl-F), receive until handshaking code is detected. If it is the end-record, close the file, else goto Step 2. If "char-Z" is NACK (Ctrl-U), receive until handshaking code is detected. If the same record is sent 5 times, abort download, else goto step 3. Else display "char-Z", receive and display until handshake code appears on the screen. Abort download.	 :For next record. ;Retry. :Error message from MICE.

Appendix E

NEW MICE Driver Programs

USD (Universal Symbolic Debugger) available on various host computer and its medium:

	Host Computer	Standard Medium
USD/PC	IBM/PC/XT/AT	5-1/4" DSDD floppy diskette
USD/PC	IBM PS/2	3-1/2" 2S/HD floppy diskette
USD/9801	NEC PC-9801 family	5-1/4" DSDD floppy diskette
USD/VMS	VAX	TU-80 Magtape, Files-11 format
USD/VMS	Micro VAX	TK-50 Cartridge tape, Files-11 format
USD/ULTRIX	Micro VAX	TK-50 Cartridge tape, Tar format
USD/SUN	SUN microsystems	1/4" Cartridge tape, Tar format

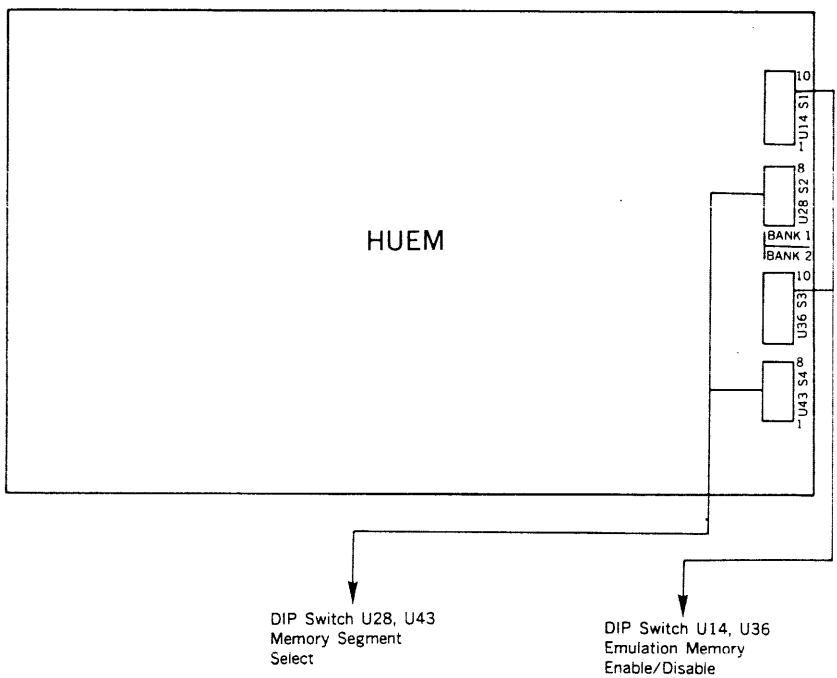
Appendix F

High Performance Universal Emulation Memory Board (HUEM)

The High Performance Universal Emulation Memory (HUEM) board is the bottom board in the conventional three board NEW MICE-II module. Each memory board supports 128K bytes of emulation memory.

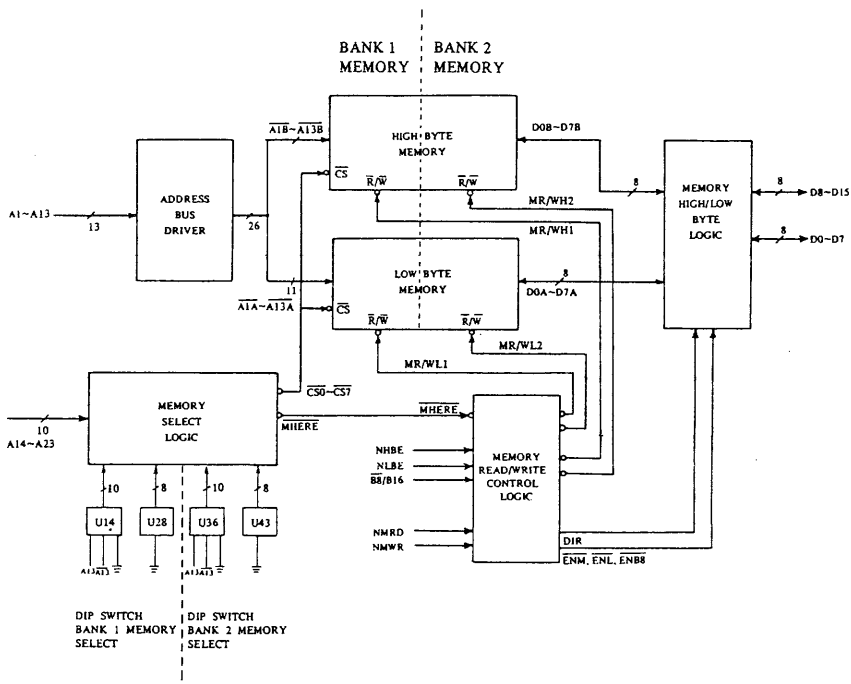
Appendix F

High Performance Universal Emulation Memory Board (HUEM) Placement Chart



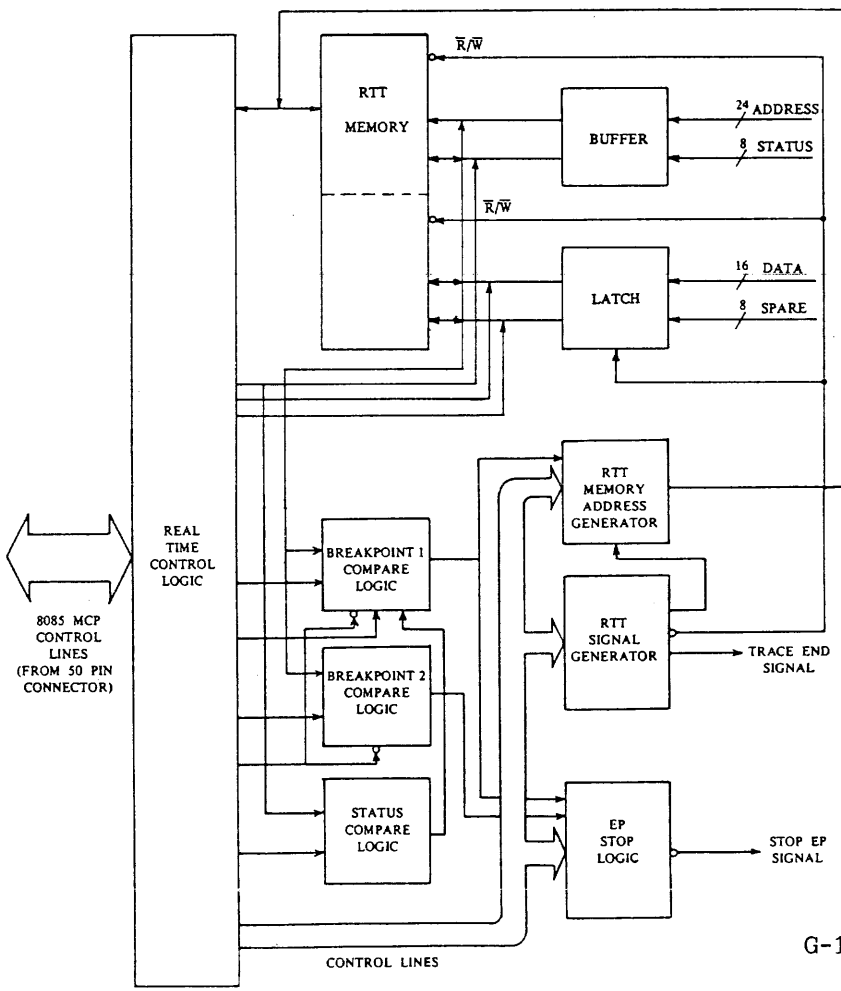
Appendix F

High Performance Universal Emulation Memory Board (HUEM) Block Diagram



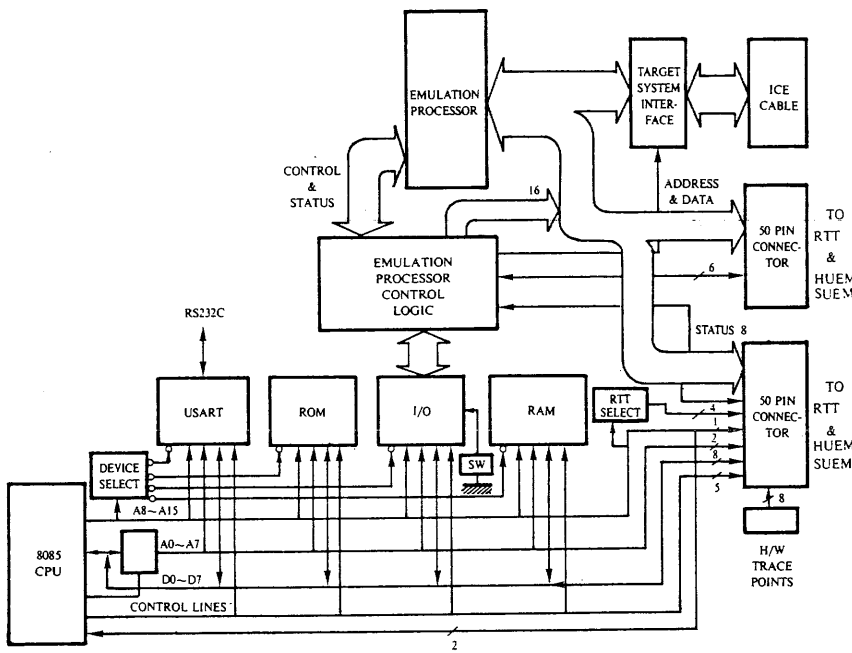
Appendix G

Real-time Trace Board (RTT) Block Diagram



Appendix H

Control Emulation Processor Board (CEP) Block Diagram

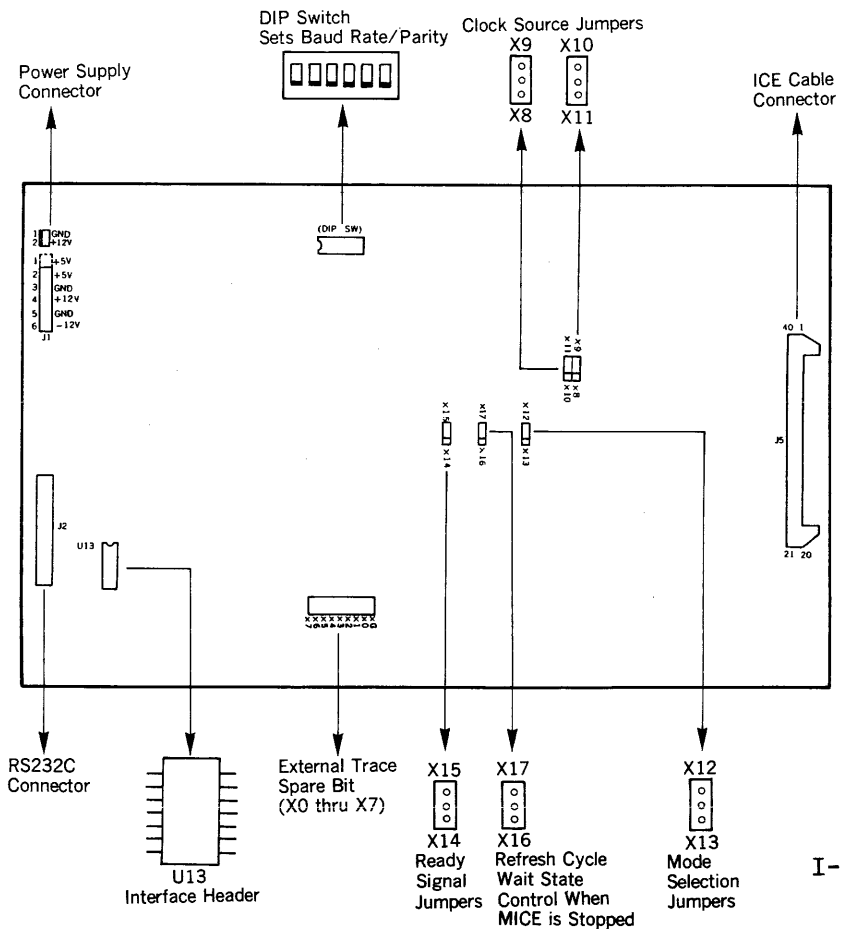


Appendix I

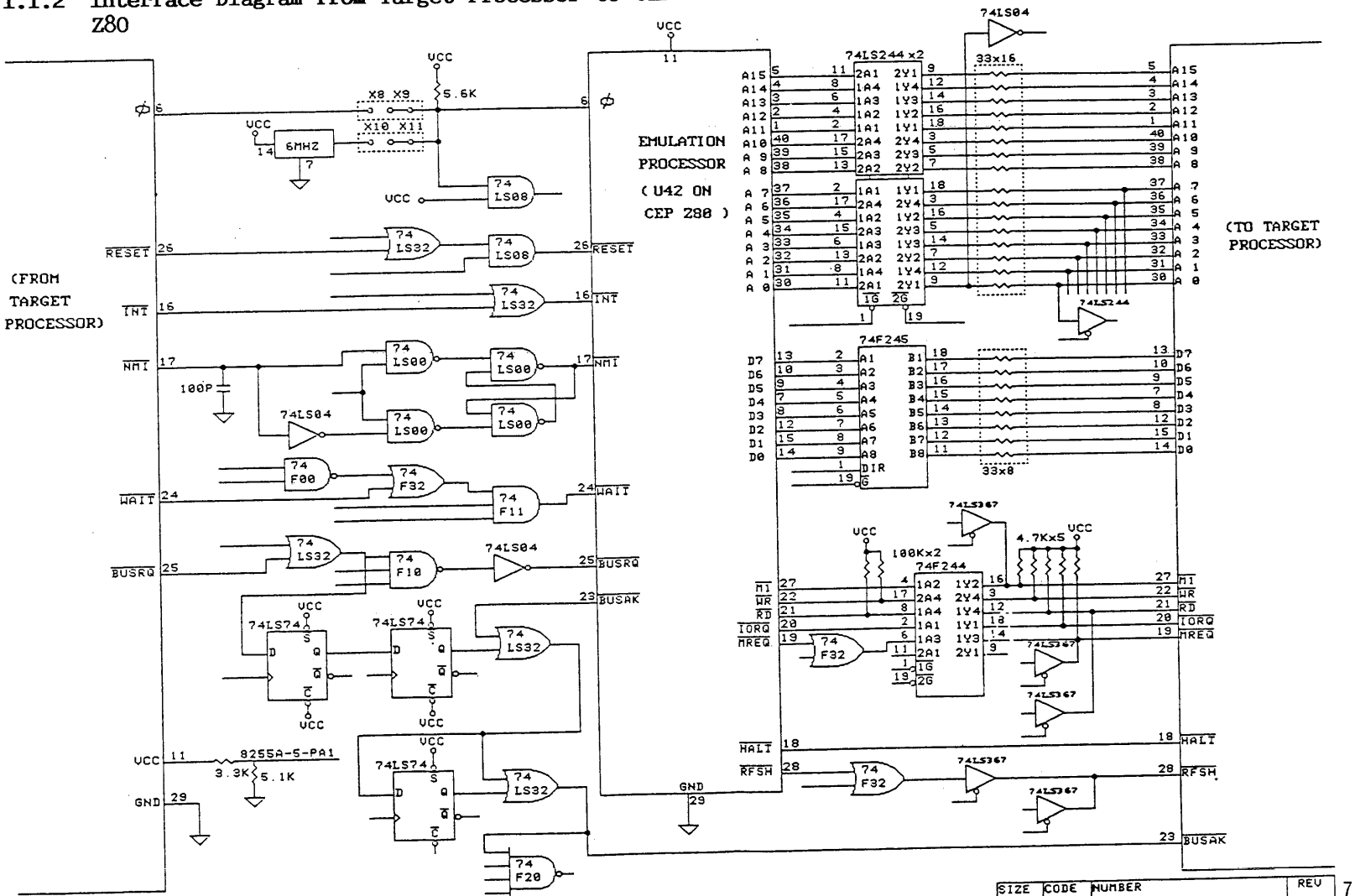
Z80

I.1 Hardware

I.1.1 Control Emulation Processor Board, CEP-Z80



I.1.2 Interface Diagram from Target Processor to CEP-Z80



SIZE	CODE	NUMBER	REV
B		140112-303	B
DATE 7/7/86		SHEET 2 OF 2	

I.2 Application Notes for the Z80 Command Set

1. If a trace or breakpoint is set to stop at a memory write cycle, NEW MICE-II Z80 will stop at the next cycle.

2. For Z80 peripheral chips which use the RETI instruction to reset interrupt requests or internal flags, the instruction must reside in target system memory (not in emulation memory).

3. When executing a RETI instruction (whose opcodes are ED and 4D) while in BUSAK mode, the peripheral does not allow the CPU to stop in the middle of these two bytes (ED and 4D).

I.3 Assembly Mnemonic Code Summary, with Disassembly Examples

LOC	OBJ	LINE	LABEL	SOURCE CODE	
0000	ED5A	0001	S0000	ADC	HL,DE
0002	CE00	0002		ADC	A,00
0004	87	0003		ADD	A,A
0005	C600	0004		ADD	A,00
0007	86	0005		ADD	A,(HL)
0008	DD8600	0006		ADD	A,(IX+00)
000B	FD8600	0007		ADD	A,(IY+00)
000E	19	0008		ADD	HL,DE
000F	DD19	0009		ADD	IX,DE
0011	FD19	0010		ADD	IY,DE
0013	E600	0011		AND	00
0015	CB4B	0012		BIT	1,E
0017	CB4E	0013		BIT	1,(HL)
0019	DDCB004E	0014		BIT	1,(IX+00)
001D	FDCB094E	0015		BIT	1,(IY+09)
0021	CD0000	0016		CALL	0000
0024	E40000	0017		CALL	PO,0000
0027	3F	0018		CCF	
0028	FE00	0019		CP	00
002A	EDA9	0020		CPD	
002C	EDB9	0021		CPDR	
002E	EDA1	0022		CPI	
0030	EDB1	0023		CPIR	
0032	2F	0024		CPL	
0033	27	0025	B0033	DAA	
0034	1B	0026		DEC	DE
0035	DD2B	0027		DEC	IX
0037	FD2B	0028		DEC	IY
0039	05	0029		DEC	B
003A	F3	0030		DI	

```
LOC     OBJ         LINE     LAB      SOURCE CODE
003B    10C3        0031             DJNZ  0000
003D    FB          0032             EI
003E    EB          0033             EX    DE,HL
003F    08          0034             EX    AF,AF'
0040    E3          0035             EX    (SP),HL
0041    DDE3        0036             EX    (SP),IX
0043    FDE3        0037             EX    (SP),IY
0045    D9          0038             EXX
0046    76          0039             HALT
0047    ED46        0040             IM    0
0049    ED56        0041             IM    1
004B    ED5E        0042             IM    2
004D    DBA0        0043             IN    A,(A0)
004F    ED78        0044             IN    A,(C)
0051    3C          0045             INC   A
0052    34          0046             INC   (HL)
0053    DD3409      0047             INC   (IX+09)
0056    FD3407      0048             INC   (IY+07)
0059    14          0049             INC   D
005A    DD23        0050             INC   IX
005C    FD23        0051             INC   IY
005E    EDAA        0052             IND
0060    EDBA        0053             INDR
0062    EDA2        0054             INI
0064    EDB2        0055             INIR
0066    E9          0056             JP    (HL)
0067    DDE9        0057             JP    (IX)
0069    FDE9        0058             JP    (IY)
006B    C30000      0059             JP    0000
006E    FA0000      0060             JP    M,0000
0071    188D        0061             JR    0000
0073    388B        0062             JR    C,0000
0075    30FF        0063             JR    NC,0076
0077    28BA        0064     B0077   JR    Z,0033
0079    20FC        0065             JR    NZ,0077
```

```
LOC     OBJ         LINE     LABEL   SOURCE CODE
007B    51          0066             LD     D,C
007C    1699        0067             LD     D,99
007E    66          0068             LD     H,(HL)
007F    DD4E07      0069             LD     C,(IX+07)
0082    FD5606      0070             LD     D,(IY+06)
0085    210D00      0071             LD     HL,000D
0088    DD7707      0072             LD     (IX+07),A
008B    FD7106      0073             LD     (IY+06),C
008E    3602        0074             LD     (HL),02
0090    DD360643    0075             LD     (IX+06),43
0094    FD360677    0076             LD     (IY+06),77
0098    0A          0077             LD     A,(BC)
0099    1A          0078             LD     A,(DE)
009A    3A6500      0079             LD     A,(0065)
009D    02          0080             LD     (BC),A
009E    12          0081             LD     (DE),A
009F    326600      0082             LD     (0066),A
00A2    ED57        0083             LD     A,I
00A4    ED5F        0084             LD     A,R
00A6    ED47        0085             LD     I,A
00A8    ED4F        0086             LD     R,A
00AA    118800      0087             LD     DE,0088
00AD    DD218800    0088             LD     IX,0088
00B1    FD214400    0089             LD     IY,0044
00B5    2A4400      0090             LD     HL,(0044)
00B8    ED4B4400    0091             LD     BC,(0044)
00BC    DD2A7600    0092             LD     IX,(0076)
00C0    FD2A7500    0093             LD     IY,(0075)
00C4    227600      0094             LD     (0076),HL
00C7    ED535400    0095             LD     (0054),DE
00CB    DD224400    0096             LD     (0044),IX
00CF    FD226600    0097             LD     (0066),IY
00D3    F9          0098             LD     SP,HL
00D4    DDF9        0099             LD     SP,IX
00D6    FDF9        0100             LD     SP,IY
```

```
LOC   OBJ        LINE   LABEL   SOURCE CODE
00D8  EDA8       0101            LDD
00DA  EDB8       0102            LDDR
00DC  EDA0       0103            LDI
00DE  EDB0       0104            LDIR
00E0  ED44       0105            NEG
00E2  00         0106            NOP
00E3  B7         0107            OR      A
00E4  EDBB       0108            OTDR
00E6  EDB3       0109            OTIR
00E8  D307       0110            OUT     (07),A
00EA  ED79       0111            OUT     (C),A
00EC  EDAB       0112            OUTD
00EE  EDA3       0113            OUTI
00F0  D1         0114            POP     DE
00F1  DDE1       0115            POP     IX
00F3  FDE1       0116            POP     IY
00F5  D5         0117            PUSH    DE
00F6  DDE5       0118            PUSH    IX
00F8  FDE5       0119            PUSH    IY
00FA  CB8F       0120            RES     1,A
00FC  C9         0121            RET
00FD  F8         0122            RET     M
00FE  ED4D       0123            RETI
0100  ED45       0124            RETN
0102  CB17       0125            RL      A
0104  17         0126            RLA
0105  CB01       0127            RLC     C
0107  CB06       0128            RLC     (HL)
0109  DDCB0706   0129            RLC     (IX+07)
010D  FDCB0706   0130            RLC     (IY+07)
0111  07         0131            RLCA
0112  ED6F       0132            RLD
0114  CB1F       0133            RR      A
0116  1F         0134            RRA
0117  CB0A       0135            RRC     D
```

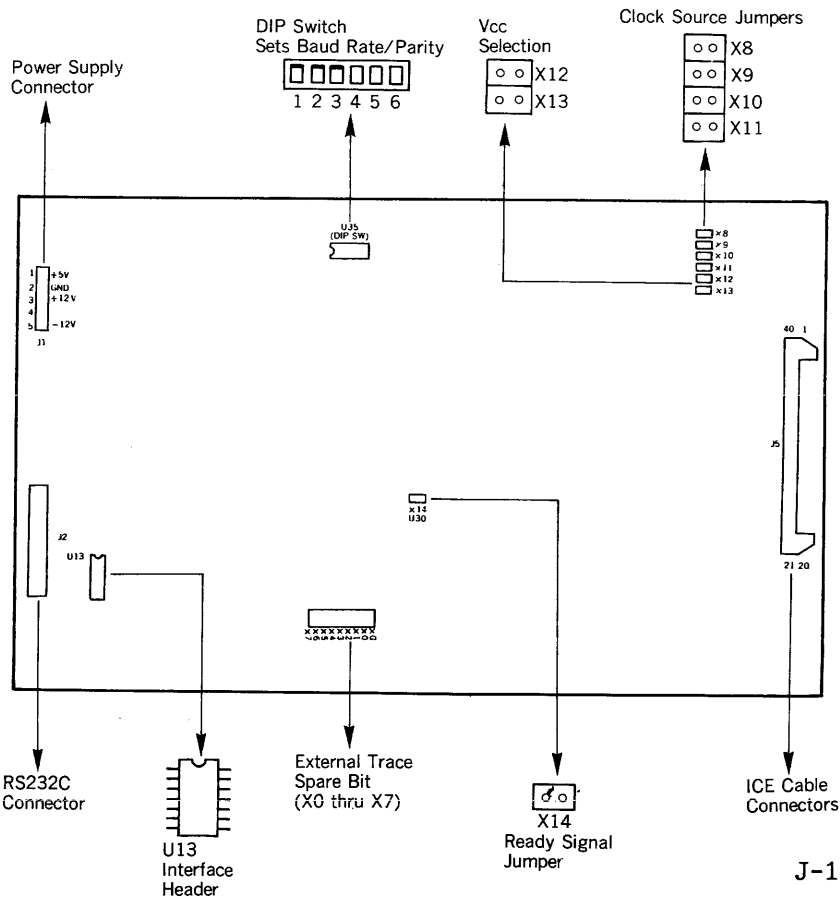
```
LOC     OBJ         LINE    LABEL   SOURCE CODE
0119    0F          0136            RRCA
011A    ED67        0137            RRD
011C    C7          0138            RST   00
011D    98          0139            SBC   A,B
011E    ED52        0140            SBC   HL,DE
0120    37          0141            SCF
0121    CBCF        0142            SET   1,A
0123    CBCE        0143            SET   1,(HL)
0125    DDCB07CE    0144            SET   1,(IX+07)
0129    FDCB67CE    0145            SET   1,(IY+67)
012D    CB20        0146            SLA   B
012F    CB2B        0147            SRA   E
0131    CB3A        0148            SRL   D
0133    90          0149            SUB   B
0134    AF          0150            XOR   A
```

Appendix J

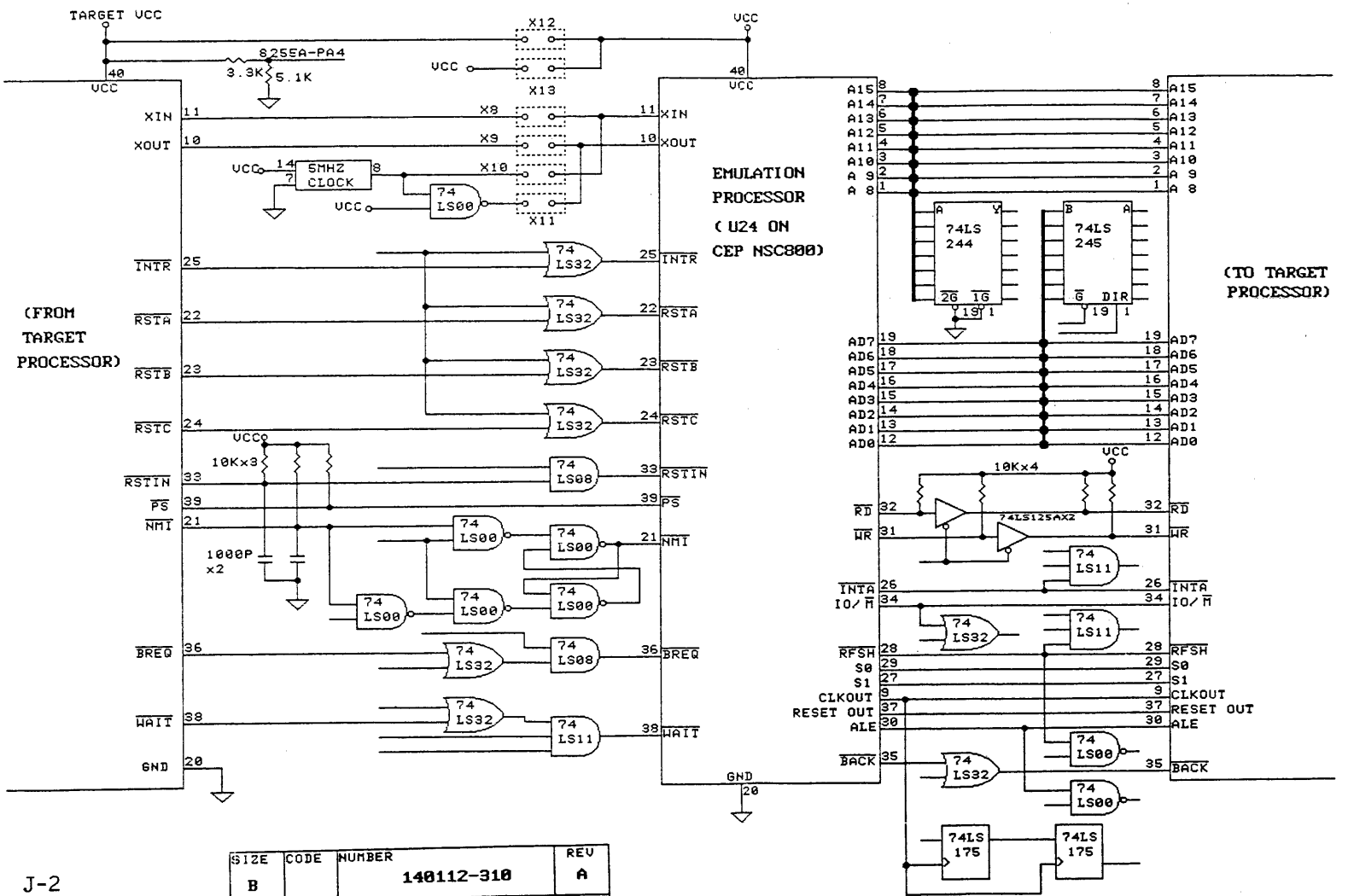
NSC800

J.1 Hardware

J.1.1 Control Emulation Processor Board, CEP-NSC800



J.1.2 Interface Diagram from Target Processor to CEP-NSC800



SIZE	CODE	NUMBER	REV
B		140112-310	A
DATE		SHEET 2 OF 2	

J.2 Application Notes for NSC800 Command Set

1. During I/O read and write cycles, the I/O port address is duplicated in the upper byte of the address. If port addresses are used as halt or trace addresses, the command specification must also duplicate the port address in the upper byte.

2. Dynamic memory refresh from the NSC800 is not supported by NEW MICE-II NSC800 during Cycle Step or Instruction Step operation.

J.3 Assembly Mnemonic Code Summary, with Disassembly Examples

Refer to section I.3.

Appendix K

Z8

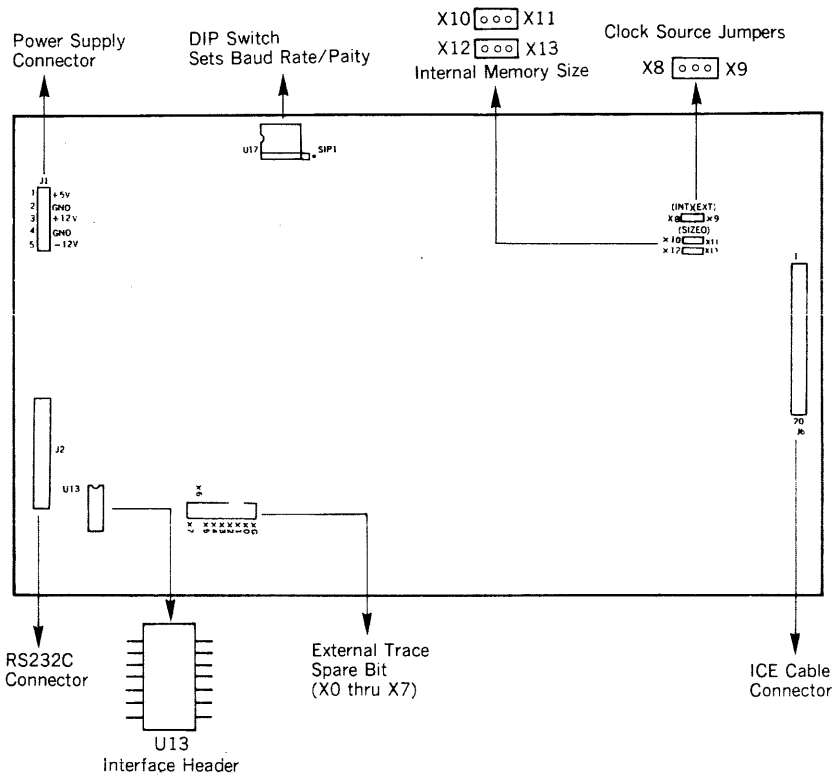
K.1 Hardware

NEW MICE-II Z8 comes with two or three boards, depending on the user's specified emulation requirements. When only internal memory is used, the CEP-Z8 and RTT boards are all that is required. However, when external memory is used, SUEM or HUEM is also required. When replacing the CEP board on another MICE model with the CEP-Z8, SUEM/HUEM may be removed if desired.

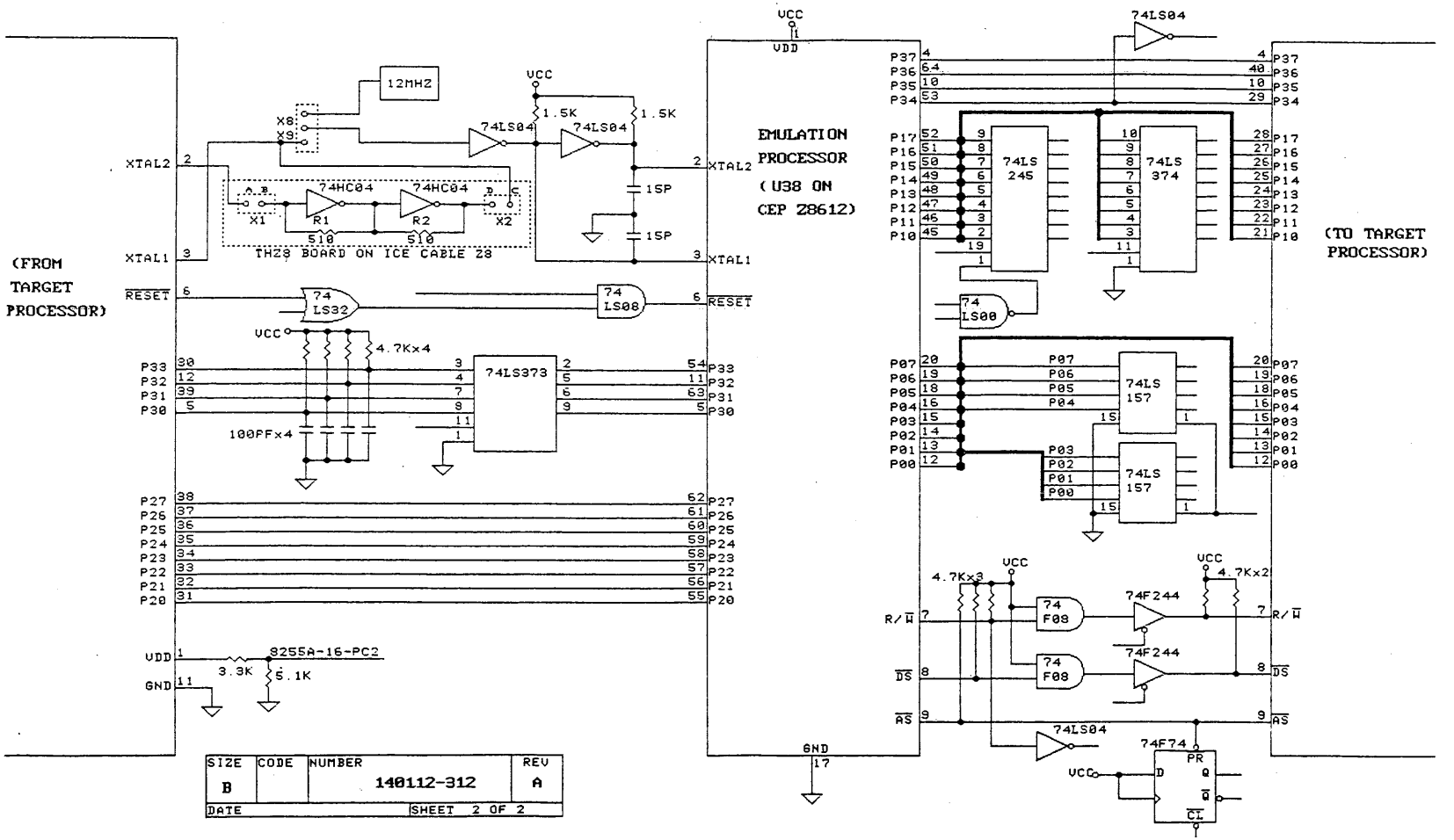
1. The CEP-Z8 uses a Z8612 to emulate Z8611, Z8613 and Z8681 processors.

2. Z8601, Z8603, Z8671, Z8681 and Z8682 processors can also be emulated; but external program memory (2K-4K bytes for Z8601, Z8603, Z8671, Z8682, or 0K-4K bytes for Z8681). must be transferred to the CEP-Z8 RAM at U51(6264). Either read the data into the host computer and then download it to MICE emulation memory, or place the target ROM on the CEP-Z8 at U51. However, note that data memory (2K-4K bytes for Z8601, Z8603, Z8671, Z8682, or 0K-4K bytes for Z8681) cannot be accessed by the user program.

K.1.1 Control Emulation Processor Board, CEP-Z8



K.1.2 Interface Diagram from Target Processor to CEP-Z8



K.2 Application Notes for the Z8 Command Set

1. The Z8 family has no HALT or READY state. When the emulation CPU is stopped by any MICE command, it will execute a "JR $" instruction to prevent the program counter from changing to another location.

2. If execution is halted by H3/H4 at a one-byte instruction at the last cycle of a multi-byte instruction, the program counter will equal the halt address plus two instructions. For any other break location, the program counter will equal the halt address plus one instruction. If any other breakpoint is encountered, or the program is cycle-stepped or in-struction-stepped, the program counter will point to next instruction.

3. When port 1 is set at high impedance mode (i.e. D3 & D4 = 1 in RF8 [RF8 = XXX11XXX]), MICE control program will overwrite one register during A/I/J/M/O/R/SR/T/Z, Upload and Download commands. The user should define any general purpose register R4_R7F to be overwritten, otherwise default is for R4.

   Example: If RF8(P01M) = 9EH, then RW will display as indicated below.

```
>RW
*** HIGH-IMPEDANCE AD0-AD7,AS~,DS~,R/W~!
*** OVERWRITE WHICH REGISTER (04H_7FH, <CR>=04H)? 3F
RP   0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
20  0A 31 0C E3 00 BF 46 23 23 23 23 78 DA 0E 23 23
>
```

4. If PC(H/L) and FLAGS are saved at R04H-R06H on the stack, and an internal interrupt is accepted during A/I/J/M/O/R/SR/T/Z, Upload or Download command execution, the PC and FLAGS will be overwritten.

5. If P0/P1 are both set to I/O mode, and R04$\geq$10H, and the register pointer (RP) is within an unused range (i.e. 80H-E0H), the contents of RFEH and RFFH will be overwritten during A/I/J/M/O/R/SR/T/Z, Upload or Download command execution.

K.3 Z8 Assembly Mnemonic Code Summary, with Disassembly Examples

```
LOC     OBJ       LINE     LABEL     SOURCE CODE
0004    140100    0003               ADC   00,01
0007    1402E4    0004               ADC   R04,02
000A    14E503    0005               ADC   03,R05
000D    150604    0006               ADC   04,@06
0010    1507E6    0007               ADC   R06,@07
0013    15E708    0008               ADC   08,@R07
0016    160901    0009               ADC   09,#01
0019    16E823    0010               ADC   R08,#23
001C    170A45    0011               ADC   @0A,#45
001F    17E967    0012               ADC   @R09,#67
0022    02AB      0013               ADD   R0A,R0B
0024    03CD      0014               ADD   R0C,@R0D
0026    040C0B    0015               ADD   0B,0C
0029    040DEE    0016               ADD   R0E,0D
002C    04EF0E    0017               ADD   0E,R0F
002F    05100F    0018               ADD   0F,@10
0032    0511E0    0019               ADD   R00,@11
0035    05E112    0020               ADD   12,@R01
0038    061389    0021               ADD   13,#89
003B    06E2AB    0022               ADD   R02,#AB
003E    0714CD    0023               ADD   @14,#CD
0041    07E3EF    0024               ADD   @R03,#EF
0044    5245      0025               AND   R04,R05
0046    5367      0026               AND   R06,@R07
0048    541615    0027               AND   15,16
004B    5417E8    0028               AND   R08,17
004E    54E918    0029               AND   18,R09
0051    551A19    0030               AND   19,@1A
0054    551BEA    0031               AND   R0A,@1B
0057    55EB1C    0032               AND   1C,@R0B
005A    561D41    0033               AND   1D,#41
005D    56EC42    0034               AND   R0C,#42
```

LOC	OBJ	LINE	LABEL	SOURCE CODE	
0060	571E43	0035		AND	@1E,#43
0063	57ED44	0036		AND	@ROD,#44
0066	D60012	0037		CALL	$0012
0069	D420	0038		CALL	@20
006B	D4EE	0039		CALL	@RROE
006D	EF	0040		CCF	
006E	B021	0041		CLR	21
0070	B0EF	0042		CLR	ROF
0072	B122	0043		CLR	@22
0074	B1E0	0044		CLR	@R00
0076	6023	0045		COM	23
0078	60E1	0046	B0078	COM	R01
007A	6124	0047		COM	@24
007C	61E2	0048		COM	@R02
007E	A234	0049		CP	R03,R04
0080	A356	0050		CP	R05,@R06
0082	A42625	0051		CP	25,26
0085	A427E7	0052		CP	R07,27
0088	A4E828	0053		CP	28,R08
008B	A52A29	0054		CP	29,@2A
008E	A52BE9	0055		CP	R09,@2B
0091	A5EA2C	0056		CP	2C,@R0A
0094	A62DFE	0057		CP	2D,#FE
0097	A6EBDC	0058		CP	R0B,#DC
009A	A72EBA	0059	B009A	CP	@2E,#BA
009D	A7EC98	0060		CP	@R0C,#98
00A0	402F	0061		DA	2F
00A2	40ED	0062		DA	R0D
00A4	4130	0063		DA	@30
00A6	41EE	0064		DA	@R0E
00A8	0031	0065		DEC	31
00AA	00EE	0066	B00AA	DEC	R0E
00AC	0132	0067		DEC	@32
00AE	01EF	0068		DEC	@R0F
00B0	8032	0069		DECW	32

```
LOC   OBJ      LINE   LABEL   SOURCE CODE
00B2  80E0     0070           DECW  RR00
00B4  8133     0071           DECW  @33
00B6  81E1     0072           DECW  @R01
00B8  8F       0073           DI
00B9  2A7F     0074           DJNZ  R02,$013A
00BB  3A80     0075           DJNZ  R03,$003D
00BD  9F       0076           EI
00BE  4E       0077           INC   R04
00BF  2034     0078           INC   34
00C1  2135     0079           INC   @35
00C3  21E5     0080           INC   @R05
00C5  A036     0081           INCW  36
00C7  A0E6     0082           INCW  RR06
00C9  A137     0083           INCW  @37
00CB  A1E7     0084           INCW  @R07
00CD  BF       0085           IRET
00CE  0D0034   0086           JP    F,$0034
00D1  1D0056   0087           JP    LT,$0056
00D4  2D0078   0088           JP    LE,$0078
00D7  3D009A   0089           JP    ULE,$009A
00DA  4D00BC   0090           JP    OV,$00BC
00DD  5D00DE   0091           JP    MI,$00DE
00E0  6D00F0   0092           JP    EQ,$00F0
00E3  7D0100   0093           JP    ULT,$0100
00E6  8D0200   0094           JP    $0200
00E9  9D0300   0095           JP    GE,$0300
00EC  AD0400   0096           JP    GT,$0400
00EF  BD0500   0097           JP    UGT,$0500
00F2  CD0600   0098           JP    NOV,$0600
00F5  DD0700   0099           JP    PL,$0700
00F8  ED0800   0100           JP    NE,$0800
00FB  FD0900   0101           JP    UGE,$0900
00FE  0BCA     0102           JR    F,$00CA
0100  1BA8     0103   B0100   JR    LT,$00AA
0102  2B86     0104           JR    LE,$008A
```

OBJ	LINE	LABEL		SOURCE CODE	
0104	3B32	0105		JR	ULE,$0138
0106	4B33	0106		JR	OV,$013B
0108	5B3E	0107		JR	MI,$0148
010A	6B4E	0108		JR	EQ,$015A
010C	7B6E	0109		JR	ULT,$017C
010E	8B7E	0110		JR	$018E
0110	9B77	0111		JR	GE,$0189
0112	AB71	0112		JR	GT,$0185
0114	BB63	0113		JR	UGT,$0179
0116	CB55	0114		JR	NOV,$016D
0118	DB42	0115		JR	PL,$015C
011A	EB37	0116		JR	NE,$0153
011C	FB36	0117		JR	UGE,$0154
011E	8C44	0118		LD	R08,#44
0120	9839	0119		LD	R09,39
0122	A8EB	0120		LD	R0A,R0B
0124	B93A	0121		LD	3A,R0B
0126	E3CD	0122		LD	R0C,@R0D
0128	F3EF	0123		LD	@R0E,R0F
012A	E4603B	0124		LD	3B,60
012D	E53E3C	0125		LD	3C,@3E
0130	E53FE0	0126		LD	R00,@3F
0133	E5E140	0127		LD	40,@R01
0136	E64245	0128		LD	42,#45
0139	2C46	0129		LD	R02,#46
013B	E74300	0130	B013B	LD	@43,#00
013E	E7E301	0131		LD	@R03,#01
0141	F54544	0132		LD	@44,45
0144	F546E4	0133		LD	@R04,46
0147	F5E547	0134		LD	@47,R05
014A	C76748	0135		LD	R06,48(R07)
014D	C78AE9	0136		LD	R08,R09(R0A)
0150	D7CB49	0137		LD	49(R0B),R0C
0153	D7FEED	0138	B0153	LD	R0D(R0E),R0F

OBJ	LINE	LABEL		SOURCE CODE	
0156	C202	0139		LDC	R00,@RR02
0158	D254	0140		LDC	@RR04,R05
015A	C368	0141	B015A	LDCI	@R06,@RR0
015C	D3BA	0142	B015C	LDCI	@RR0A,@R0
015E	82CE	0143		LDE	R0C,@RR0E
0160	9210	0144		LDE	@RR00,R01
0162	8324	0145		LDEI	@R02,@RR0
0164	9356	0146		LDEI	@RR06,@R0
0166	FF	0147		NOP	
0167	4278	0148		OR	R07,R08
0169	439A	0149		OR	R09,@R0A
016B	444B4A	0150		OR	4A,4B
016E	444CEB	0151		OR	R0B,4C
0171	44EC4D	0152		OR	4D,R0C
0174	454F4E	0153		OR	4E,@4F
0177	4550ED	0154		OR	R0D,@50
017A	45EE51	0155		OR	51,@R0E
017D	465202	0156		OR	52,#02
0180	46EF03	0157		OR	R0F,#03
0183	475304	0158		OR	@53,#04
0186	47E005	0159		OR	@R00,#05
0189	5054	0160	B0189	POP	54
018B	50E1	0161		POP	R01
018D	5155	0162		POP	@55
018F	51E2	0163		POP	@R02
0191	7056	0164		PUSH	56
0193	70E2	0165		PUSH	R02
0195	7157	0166		PUSH	@57
0197	71E3	0167		PUSH	@R03
0199	CF	0168		RCF	
019A	AF	0169		RET	
019B	9058	0170		RL	58
019D	90E4	0171		RL	R04
019F	9159	0172		RL	@59
01A1	91E5	0173		RL	@R05

OBJ	LINE	LABEL	SOURCE CODE	
01A3	105A	0174	RLC	5A
01A5	10E6	0175	RLC	R06
01A7	115B	0176	RLC	@5B
01A9	11E7	0177	RLC	@R07
01AB	E05C	0178	RR	5C
01AD	E0E8	0179	RR	R08
01AF	E15D	0180	RR	@5D
01B1	E1E8	0181	RR	@R08
01B3	C05E	0182	RRC	5E
01B5	C0E9	0183	RRC	R09
01B7	C15F	0184	RRC	@5F
01B9	C1EA	0185	RRC	@R0A
01BB	32BC	0186	SBC	R0B,R0C
01BD	33DE	0187	SBC	R0D,@R0E
01BF	346160	0188	SBC	60,61
01C2	34FFEF	0189	SBC	R0F,FF
01C5	34E062	0190	SBC	62,R00
01C8	356463	0191	SBC	63,@64
01CB	3565E1	0192	SBC	R01,@65
01CE	35E266	0193	SBC	66,@R02
01D1	366706	0194	SBC	67,#06
01D4	36E307	0195	SBC	R03,#07
01D7	376808	0196	SBC	@68,#08
01DA	37E409	0197	SBC	@R04,#09
01DD	DF	0198	SCF	
01DE	D069	0199	SRA	69
01E0	D0E5	0200	SRA	R05
01E2	D16A	0201	SRA	@6A
01E4	D1E6	0202	SRA	@R06
01E6	310A	0203	SRP	#0A
01E8	2278	0204	SUB	R07,R08
01EA	239A	0205	SUB	R09,@R0A
01EC	246C6B	0206	SUB	6B,6C
01EF	246DEB	0207	SUB	R0B,6D
01F2	24EC6E	0208	SUB	6E,R0C

OBJ	LINE	LABEL	SOURCE CODE	
01F5	25706F	0209	SUB	6F,@70
01F8	2571ED	0210	SUB	R0D,@71
01FB	25EE72	0211	SUB	72,@R0E
01FE	26730B	0212	SUB	73,#0B
0201	26EF0C	0213	SUB	R0F,#0C
0204	27740D	0214	SUB	@74,#0D
0207	27E00E	0215	SUB	@R00,#0E
020A	F075	0216	SWAP	75
020C	F0E1	0217	SWAP	R01
020E	F176	0218	SWAP	@76
0210	F1E2	0219	SWAP	@R02
0212	6234	0220	TCM	R03,R04
0214	6356	0221	TCM	R05,@R06
0216	647877	0222	TCM	77,78
0219	6479E7	0223	TCM	R07,79
021C	64E87A	0224	TCM	7A,R08
021F	657C7B	0225	TCM	7B,@7C
0222	657DE9	0226	TCM	R09,@7D
0225	65EA7E	0227	TCM	7E,@R0A
0228	667F0F	0228	TCM	7F,#0F
022B	66EB10	0229	TCM	R0B,#10
022E	67F020	0230	TCM	@F0,#20
0231	67EC30	0231	TCM	@R0C,#30
0234	72DE	0232	TM	R0D,R0E
0236	73F0	0233	TM	R0F,@R00
0238	74F2F1	0234	TM	F1,F2
023B	74F3E1	0235	TM	R01,F3
023E	74E2F4	0236	TM	F4,R02
0241	74FEE3	0237	TM	R03,FE
0244	74E4F5	0238	TM	F5,R04
0247	75F7F6	0239	TM	F6,@F7
024A	75F8E5	0240	TM	R05,@F8
024D	75E6F9	0241	TM	F9,@R06
0250	76FA40	0242	TM	FA,#40
0253	76E750	0243	TM	R07,#50

OBJ	LINE	LABEL	SOURCE CODE	
0256	77FB60	0244	TM	@FB,#60
0259	77E870	0245	TM	@R08,#70
025C	B29A	0246	XOR	R09,R0A
025E	B3BC	0247	XOR	R0B,@R0C
0260	B40100	0248	XOR	00,01
0263	B402ED	0249	XOR	R0D,02
0266	B4EE03	0250	XOR	03,R0E
0269	B50504	0251	XOR	04,@05
026C	B506EF	0252	XOR	R0F,@06
026F	B5E007	0253	XOR	07,@R00
0272	B60880	0254	XOR	08,#80
0275	B6E190	0255	XOR	R01,#90
0278	B709A0	0256	XOR	@09,#A0
027B	B7E2B0	0257	XOR	@R02,#B0

Appendix L

ZS8

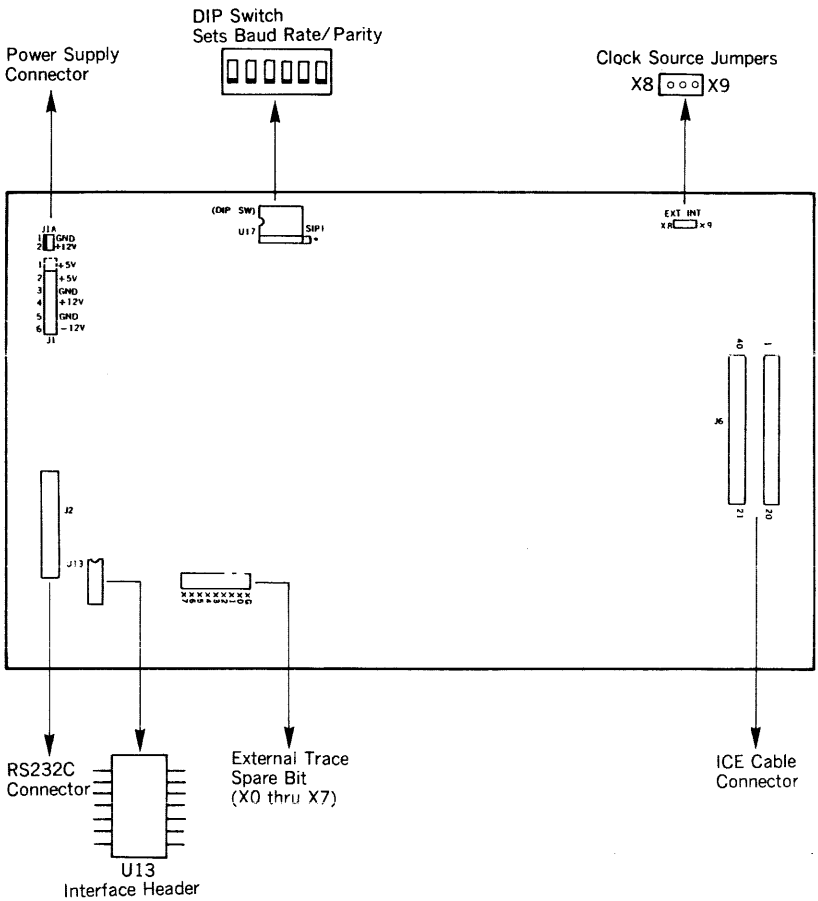
L.1 Hardware

NEW MICE-II ZS8 comes with two or three boards, depending on the user's specified emulation requirements. When only internal memory is used, the CEP-ZS8 and RTT boards are all that is required. However, when external memory is used, HUEM is also required. When replacing the CEP board on another MICE model with the CEP-ZS8, SUEM/HUEM may be removed if desired.

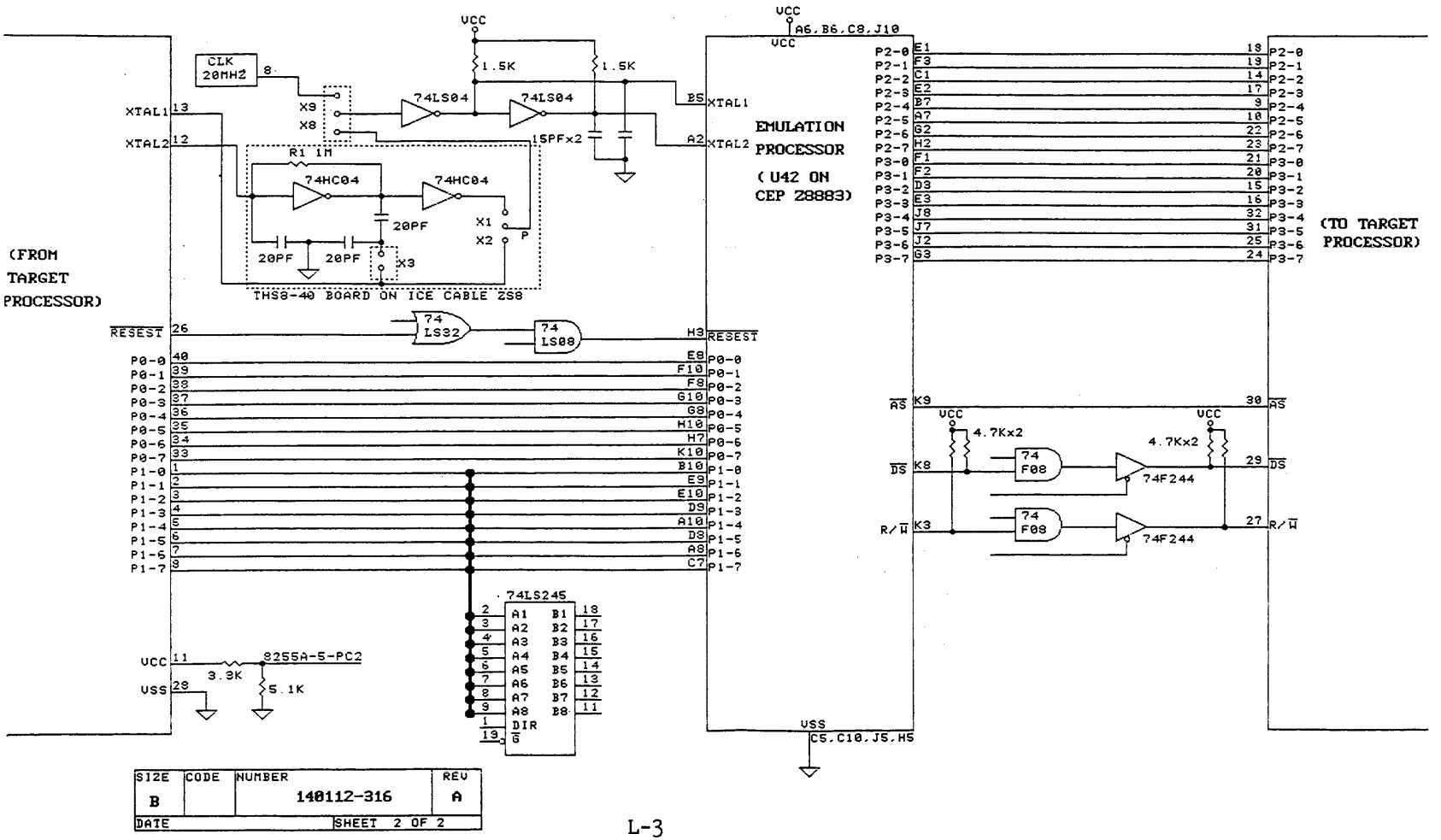
The CEP-ZS8 uses a Z8883 to emulate the following processors:

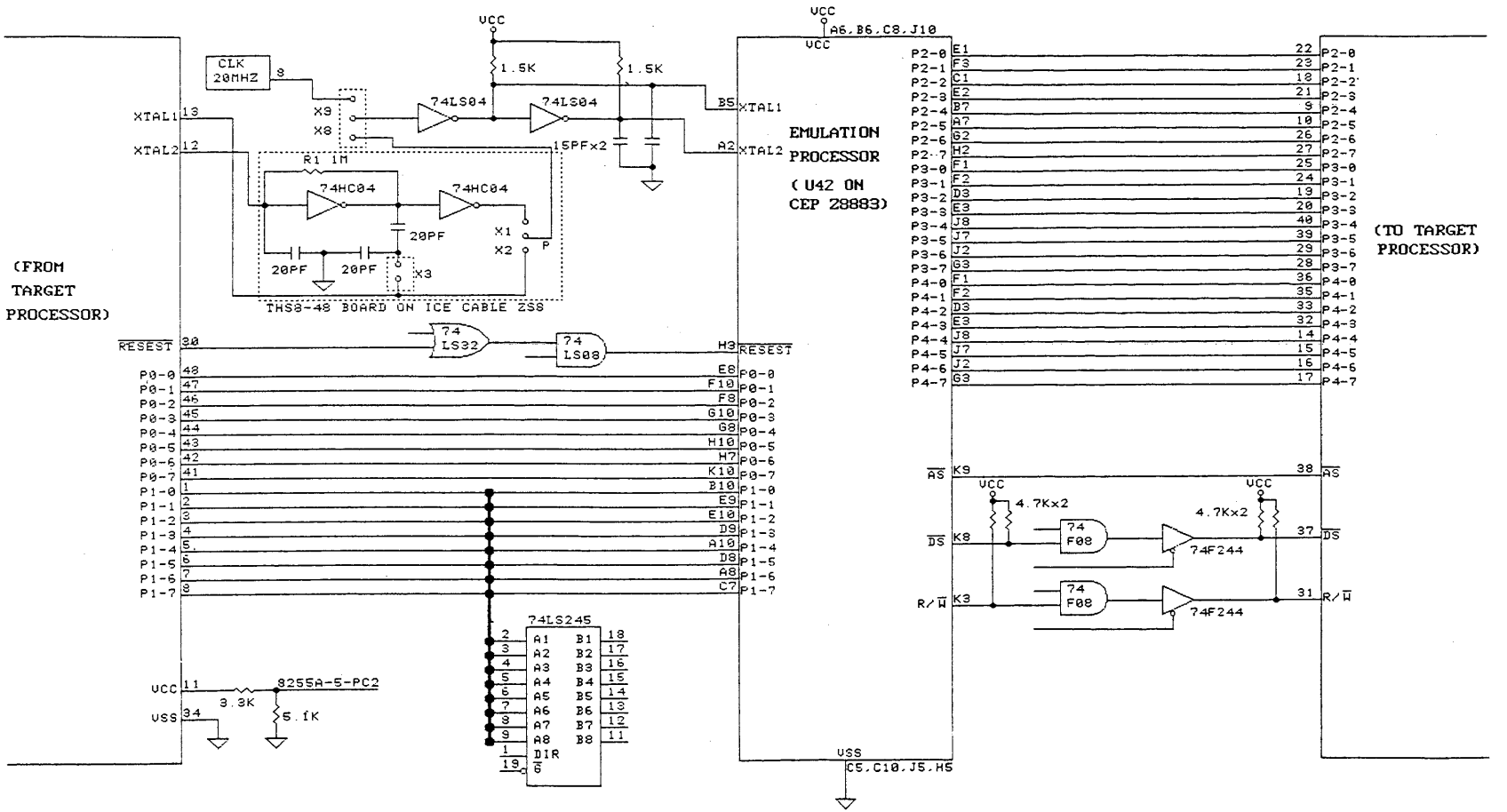
Z8800	Z8811	Z8820	Z8823	Z8832
Z8801	Z8812	Z8821	Z8830	Z8833
Z8810	Z8813	Z8822	Z8831	

L.1.1 Control Emulation Processor Board, CEP-ZS8



L.1.2 Interface Diagram from Target Processor to CEP-ZS8





SIZE	CODE	NUMBER	REV
B		140112-317	A
DATE		SHEET 2 OF 2	

L.2 Application Notes for the ZS8 Command Set

1. The ZS8 family has no HALT or READY state. When the emulation CPU is stopped by any MICE command, it will execute a "JR $" instruction to prevent the program counter from changing to another location.

2. If execution is halted by H3/H4 at a one-byte instruction at the last cycle of a multi-byte instruction, the program counter will equal the halt address plus two instructions. For any other break location, the program counter will equal the halt address plus one instruction. If any other breakpoint is encountered, or the program is cycle-stepped or instruction-stepped, the program counter will point to next instruction.

L.3 ZS8 Assembly Mnemonic Code Summary, with Disassembly Examples

```
LOC     OBJ       LINE     LABEL     SOURCE CODE
0020    1201      0001               ADC    R00,R01
0022    1323      0002               ADC    R02,@R03
0024    1403C5    0003               ADC    R05,03
0027    1507C6    0004               ADC    R06,@07
002A    16C7FF    0005               ADC    R07,#FF
002D    03AB      0006               ADD    R0A,@R0B
002F    04CD0C    0007               ADD    0C,R0D
0032    050E0D    0008               ADD    0D,@0E
0035    061012    0009               ADD    10,#12
0038    0201      0010               ADD    R00,R01
003A    0323      0011               ADD    R02,@R03
003C    04C514    0012               ADD    14,R05
003F    051716    0013               ADD    16,@17
0042    061912    0014               ADD    19,#12
0045    67801A    0015               BAND   R08,1A,#0
0048    6782C9    0016               BAND   R08,R09,#1
004B    67C7CB    0017               BAND   R0B,#3,R0C
004E    17D81C    0018               BCP    R0D,1C,#4
0051    17EACF    0019               BCP    R0E,R0F,#5
0054    570C      0020               BITC   R00,#6
0056    771E      0021               BITR   R01,#7
0058    7721      0022               BITS   R02,#0
005A    07321D    0023               BOR    R03,1D,#1
005D    0744C5    0024               BOR    R04,R05,#2
0060    07671E    0025               BOR    1E,#3,R06
0063    0789C7    0026               BOR    R07,#4,R08
0066    379AFD    0027     B0066     BTJRF  $0066,R09,#5
0069    37AD00    0028               BTJRT  $006C,R0A,#6
006C    27BE1F    0029     B006C     BXOR   R0B,1F,#7
006F    27C0CD    0030               BXOR   R0C,R0D,#0
```

```
LOC     OBJ       LINE     LABEL     SOURCE  CODE
0072    27E320    0031               BXOR    20,#1,R0E
0075    2705CF    0032               BXOR    R0F,#2,R00
0078    F60000    0033               CALL    $0000
007B    F422      0034               CALL    @22
007D    D440      0035               CALL    #$40
007F    EF        0036               CCF
0080    B0C1      0037               CLR     R01
0082    6025      0038               COM     25
0084    60C2      0039               COM     R02
0086    6126      0040               COM     @26
0088    A234      0041               CP      R03,R04
008A    A356      0042               CP      R05,@R06
008C    A42827    0043               CP      27,28
008F    A4C82A    0044               CP      2A,R08
0092    A52C2B    0045               CP      2B,@2C
0095    A6CA56    0046     B0095     CP      R0A,#56
0098    C2CBFA    0047               CPIJE   R0B,@R0C,$0095
009B    D2ED00    0048               CPIJNE  R0D,@R0E,$009E
009E    402F      0049     B009E     DA      2F
00A0    40CF      0050               DA      R0F
00A2    4130      0051               DA      @30
00A4    0031      0052               DEC     31
00A6    00C0      0053               DEC     R00
00A8    0132      0054               DEC     @32
00AA    80C2      0055               DECW    RR02
00AC    943736    0056               DIV     36,37
00AF    94C53A    0057               DIV     3A,R05
00B2    94C7C6    0058               DIV     RR06,R07
00B5    953EC8    0059               DIV     RR08,@3E
00B8    969ACA    0060               DIV     RR0A,#9A
00BB    BA00      0061               DJNZ    R0B,$00BD
00BD    20C3      0062     B00BD     INC     R03
00BF    A042      0063               INCW    42
00C1    A0CE      0064               INCW    RR0E
00C3    A143      0065               INCW    @43
```

```
LOC     OBJ         LINE   LABEL    SOURCE  CODE
00C5    BF          0066            IRET
00C6    1D4321      0067            JP      LT,$4321
00C9    7D1111      0068            JP      ULT,$1111
00CC    8D3333      0069            JP      $3333
00CF    9D2233      0070            JP      GE,$2233
00D2    FD8899      0071            JP      UGE,$8899
00D5    3044        0072            JP      @44
00D7    6B00        0073            JR      EQ,$00D9
00D9    EB00        0074    B00D9   JR      NE,$00DB
00DB    18C2        0075    B00DB   LD      R01,R02
00DD    C745        0076            LD      R04,@R05
00DF    E44847      0077            LD      47,48
00E2    984A        0078            LD      R09,4A
00E4    E54C4B      0079            LD      4B,@4C
00E7    D64EF0      0080            LD      @4E,#F0
00EA    F5504F      0081            LD      @4F,50
00ED    87DFCE      0082            LD      R0D,CE(R0F)
00F0    9743C2      0083            LD      C2(R03),R04
00F3    4768C7      0084            LDB     R06,R07,#4
00F6    47ADC9      0085            LDB     R09,#6,R0A
00F9    C3DF        0086            LDE     R0D,@RR0E
00FB    D333        0087            LDE     @RR02,R03
00FD    E74610      0088            LDC     R04,$10(RR06)
0100    E77920      0089            LDE     R07,$20(RR08)
0103    A7DE3412    0090            LDC     R0D,$1234(RR0E)
0107    A7F37856    0091            LDE     R0F,$5678(RR02)
010B    A7700010    0092            LDC     R07,$1000
010F    A7810020    0093            LDE     R08,$2000
0113    E2BC        0094            LDCD    R0B,@RR0C
0115    E2DF        0095            LDED    R0D,@RR0E
0117    E3F0        0096            LDCI    R0F,@RR00
0119    E313        0097            LDEI    R01,@RR02
011B    F234        0098            LDCPD   @RR04,R03
011D    F257        0099            LDEPD   @RR06,R05
011F    F378        0100            LDCPI   @RR08,R07
```

LOC	OBJ	LINE	LABEL	SOURCE	CODE
0121	F39B	0101		LDEPI	@RR0A,R09
0123	C4CE5A	0102		LDW	5A,RR0E
0126	C4C2C0	0103		LDW	RR00,RR02
0129	C55D5C	0104		LDW	5C,@5D
012C	C6C60020	0105		LDW	RR06,#0020
0130	846162	0106		MULT	62,61
0133	84CBCA	0107		MULT	RR0A,R0B
0136	8568CC	0108		MULT	RR0C,@68
0139	86306A	0109		MULT	6A,#30
013C	0F	0110		NEXT	
013D	42F0	0111		OR	R0F,R00
013F	4312	0112		OR	R01,@R02
0141	456FC5	0113		OR	R05,@6F
0144	46C660	0114		OR	R06,#60
0147	50C7	0115		POP	R07
0149	5172	0116		POP	@72
014B	9275C8	0117		POPUD	R08,@75
014E	9378C9	0118		POPUI	R09,@78
0151	7079	0119		PUSH	79
0153	827DCB	0120		PUSHUD	@7D,R0B
0156	837E7F	0121		PUSHUI	@7E,7F
0159	CF	0122		RCF	
015A	9081	0123		RL	81
015C	10CE	0124		RLC	R0E
015E	1184	0125		RLC	@84
0160	E085	0126		RR	85
0162	C0C0	0127		RRC	R00
0164	C188	0128		RRC	@88
0166	4F	0129		SB0	
0167	5F	0130		SB1	
0168	3334	0131		SBC	R03,@R04
016A	34C58B	0132		SBC	8B,R05
016D	358F8E	0133		SBC	8E,@8F
0170	369170	0134		SBC	91,#70
0173	DF	0135		SCF	

LOC	OBJ	LINE	LABEL	SOURCE	CODE
0174	D0C8	0136		SRA	R08
0176	D193	0137		SRA	@93
0178	3190	0138		SRP	#90
017A	31A2	0139		SRP0	#A0
017C	31B1	0140		SRP1	#B0
017E	23BC	0141		SUB	R0B,@R0C
0180	24CEB3	0142		SUB	B3,R0E
0183	25C0CF	0143		SUB	R0F,@C0
0186	26C2D0	0144		SUB	R02,#D0
0189	F0D4	0145		SWAP	D4
018B	6367	0146		TCM	R06,@R07
018D	64D9C8	0147		TCM	R08,D9
0190	65DCDB	0148		TCM	DB,@DC
0193	66E3E0	0149		TCM	E3,#E0
0196	72CD	0150		TM	R0C,R0D
0198	76C255	0151		TM	R02,#55
019B	3F	0152		WFI	
019C	B234	0153		XOR	R03,R04
019E	B356	0154		XOR	R05,@R06
01A0	B4F3C7	0155		XOR	R07,F3
01A3	B5F6F5	0156		XOR	F5,@F6
01A6	B5F7C9	0157		XOR	R09,@F7
01A9	B6FFDD	0158		XOR	FF,#DD
01AC	B6CAEE	0159		XOR	R0A,#EE

Appendix M

Error Messages

---

1. **ADDRESS xxxx RAM ERROR!**

   Condition: Memory addressed in Memory Test (T) command failed.

   Recovery: Verify that specified address is write enabled RAM; and then see if the address bus and/or data bus are open or shorted.

2. **BACKWARD TRACE FAILS!**

   Condition: Trace buffer is empty when the breakpoint is matched, or <ESC> is input, or the processor is stopped without stepping any machine cycles.

   Recovery: See if the READY signal is always inactive, or the HOLD signal is always active. Then see is the trigger address or a specified breakpoint (H1-H6. was matched immediately.

3. **BC TABLE FULL**

   Condition: Instructions branched to and called in the disassembly command exceed 900.

4. **BUS REQUESTING!**

   Condition: Target bus requesting signal always active.

   Recovery: Verify that target has a bus request signal.

5. **COMMAND TOO LONG!**

   Condition: Instruction string in assembly command exceeds 50 characters.

6. **ERROR!**

   Condition: Command syntax error.

7. **ERROR CODE!**

   Condition: Instruction error in disassembly command.

8. **ERROR CODE, TRY AGAIN!**

   Condition: Instruction error in assembly command.

9. **FORWARD TRACE FAILS!**

   Condition: Trace buffer is empty when the breakpoint is matched, or <ESC> is input, or the processor is stopped without stepping any machine cycles.

   Recovery: See if the trigger address was invalid, or a specified breakpoint was matched before the trace started.

10. **Hx NOT SET!**

    Condition: Specified breakpoint not set prior to executing BT command.

11. **MEMORY SEARCH FAILURE!**

    Condition: Data string not found within specified address range for Memory Search (M) command.

12. **MEMORY VERIFICATION FAILURE!**

    Condition: Data cannot be written to address memory in Modify (M) command.

    Recovery: Verify that specified address is valid and write enabled.

13. **MEMORY WRITE FAILURE!**

    Condition: Data cannot be written to addressed memory in assembly command.

    Recovery: Verify that specified address is valid and write enabled.

14. **NO BPP CARD!**

    Condition: BPP commands cannot be executed without a BPP card.

15. **PROGRAM HALT!**

    Condition: Emulation processor halted by instruction.

Recovery: Use an interrupt or reset to recover; then perform a trace to see where the HALT instruction occurred.

16. **TARGET CAN'T STEP! ***

Condition: Emulation processor does not respond after applying power or inputting the software reset command "r" (see Section 2.10). There is a problem with emulation processor that prevents a program from executing; there are several possibilities which can generate this message:

1. There is no clock.
2. There is a problem with the address line.
3. Target does not support the "ready" signal.
4. There is a problem with the RTT, CEP or HUEM/SUEM board.
5. Other component failure.

Recovery: Check to be sure the emulation processor clock signal is correct and verify that the clock source selection switch is properly adjusted (see CEP Placement Chart in the appropriate appendix for switch location).

17. **TARGET IS NOT READY!**

Condition: Target memory or input/output ready signal is inactive.

Recovery: Use a valid physical location in the command specification.

18. **Ux -- FAILURE! ***

Condition: MICE selftest indicates location number for failure device.

Recovery: Contact your nearest Microtek distributor for assistance.

19. **WARNING: ONLY 6 BYTES ARE VALID**

Condition: Define byte (DB) in assembly command exceeds 6 bytes.

20. **WHAT? ***

Condition: RS-232C communication with parity or framing error.

Recovery: Check interface setting for communications at U13.

*Potentially fatal error; if problem cannot be resolved, contact your nearest Microtek distributor for assistance.

APPENDIX N

LIMITED WARRANTY; service

Microtek International Inc. ("Microtek") **warrants** NEW MICE-II (the "Product") and the user's manual for the Product, to be free from physical defects for a period of twelve (12) months from the date of the original retail purchase. This warranty applies only to the original retail purchaser who bought this Product from an authorized Microtek representative. This warranty is void if the Product is damaged by improper or abnormal use or by accident, if the Product is altered or modified in any way, or if any attempt is made to repair the Product without authorization from Microtek.

If you find a Product to be defective, contact your authorized Microtek representative or Microtek. When you receive authorization to do so, return the Product as directed; include proof of purchase and purchase date. Microtek will correct physical defects in Products under warranty at no charge to you by repairing or, at its option, replacing such Products. THIS IS THE SOLE AND EXCLUSIVE REMEDY AVAILABLE FOR BREACH OF WARRANTY OR UNDER ANY OTHER LEGAL THEORY WITH RESPECT TO MICROTEK PRODUCTS. Product repairs not covered by warranty, and Product updates, are provided at a set rate.

ALL OTHER WARRANTIES AND REPRESENTATIONS, ORAL OR WRITTEN, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, ARE EXCLUDED AND DO NOT APPLY. Except as set forth in this limited warranty,

the purchaser assumes the risk as to these Products' quality, accuracy, design, and performance.

It is solely the purchaser's responsibility to determine the suitability of these Products for each particular application. Microtek Products are in all events not suitable, and are not authorized, for use in connection with life support devices or systems, or in other devices or systems potentially injurious to life or health.

IN NO EVENT WILL MICROTEK BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY BREACH OF WARRANTY OR UNDER ANY OTHER LEGAL THEORY, even if advised of the possibility of such damages. Microtek is thus not liable for lost profits or goodwill; downtime; damage or destruction of any program, data, equipment, or other property; costs of recovering, reprogramming, or reproducing any program, data, or equipment; personal injury or loss; or any other damages.

Except as required by law, no representative, agent, or employee of Microtek is authorized to commit Microtek to warranties, representations, or obligations inconsistent with or in addition to those set forth in this limited warranty.

Please complete and return the registration section of the warranty card (provided in the packing box) to Microtek.

BPP

Section

BPP SECTION FOR

Z80
NSC800
Z8
Z8S

1st Edition
August 1988

MICROTEK INTERNATIONAL INC.
No. 6 Industry E. Road 3
Science-Based Industrial Park
Hsinchu, Taiwan 30077, R.O.C.
TEL: (035) 772155
TLX: 32169 MICROTEK
FAX: (035) 772598

TABLE OF CONTENTS

SECTION 1

INTRODUCTION

Microtek's BreakPoint Processor (**BPP**) is a comprehensive breakpoint control unit for all NEW MICE-II emulators. The **BPP** is an optional single card PCB that features sophisticated breakpoint logic. It includes up to 120 new breakpoint constructs for more flexibility in target system debug and development. The **BPP** also includes external hardware triggering and an execution interval timer. This powerful triggering system helps you to quickly solve the most difficult software and hardware bugs.

Breakpoint conditions designating precise events can be logically joined with **ARM/AND/OR** connectives, permitting you to -

* begin or end tracing
* activate another event group
* specify delayed trigger count
* initiate event counter
* use external signal as trace trigger
* send trigger signal to external device
* record execution time between two events
* break emulation

Flexible trigger constructs can be used to define single events, multiple activities or external hardware signals. These powerful triggers make it easy to specify complex processor and hardware sequences. Data breakpoints can be set for address, data, status, count and delay. While external hardware breakpoints can specify up to two triggers (each with any combination of two signals).

This breakpoint system combines a large number of trigger events, permitting you to begin target trace or break emulation on the following activities -

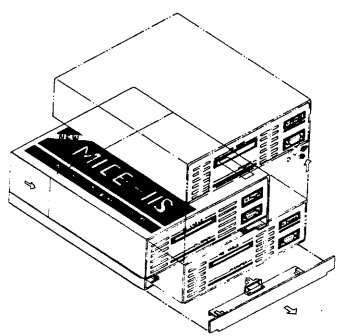
* Address Bus
* Data Bus
* Delay Count
* Event Counter (trigger match)
* Microprocessor Status Lines (Fetch, Read, Write, Input, Output and Interrupt Acknowledge)
* Logically Joined Breakpoints
* Sequence Levels
* Logical State Input

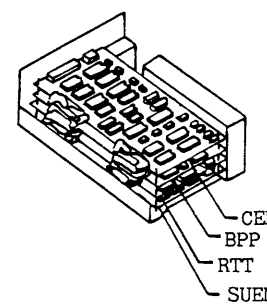
1.1 Installation

The **BPP** is a multilayer board added to the NEW MICE-II module.

To install the **BPP** board, first remove the cover to the NEW MICE-II case. With the power off, disconnect the two ribbon cable connectors; then remove the CEP and RTT boards. Replace the original brass spacers with the new ones provided with the **BPP** board. Reassemble all boards back into the MICE case, inserting the BPP board between the CEP and RTT boards. Then attach the new ribbon cable connectors (supplied with the **BPP**) firmly to each board.

Note: We recommend leaving the cover of the NEW MICE-II case off when using four boards, otherwise overheating may cause intermittent problems in operation.





1.2 Electrical and Environmental Specifications

Electrical Characteristics

Power	+5 VDC	(For BPP board)
Maximum current	1.3 Amp	

NEW MICE-II equips one built-in power supply.

Power Consumption: AC100-120V~:1.3A MAX
AC200-240V~:0.65A MAX
47-63HZ

Voltage Requirements: AC100-120V~ or
AC200-240V~
Factory-set

Environmental Characteristics

Operating temperature	0°~50°C	(32°~122°F)
Storage temperature	-10°~65°C	(14°~149°F)
Relative humidity	20-80%	

SECTION 2

BPP FUNCTIONS

The BPP includes four breakpoints (H3-H6), two triggers for transmitting a sync signal to an external device (BS1-BS2), and a Breakpoint Trigger Trace command (BT).

NEW MICE-II's other breakpoints (H1/H2) can be combined with the BPP commands; and H2-H6 can also serve as individual breakpoints for B/F/G commands (note that H1 can not be used for B/F commands).

2.1 Halt Breakpoints and Breakpoint Commands

Breakpoints:

H3/H4 are data breakpoints.

H5/H6 are external hardware breakpoints.

Commands:

BS is the Breakpoint Sync command. It outputs a sync signal (to an external device) when the specified trigger condition is matched.

BT is the Breakpoint Trace command. It begins tracing from the current program address up to the specified trigger condition (Backward Trace).

?B is the command to display BPP syntax as illustrated below:

```
>?B

H ?|[0][3|4|5|6]
H3 addx[ data[ <dbwc>][ q[ cnt]]]
H4 addx[ data[ <dbwc>][ q[ d1y]]]
H5 xx
H6 xx

BS 1|2[ addx[ data[<dbwc>][ q]]]

BT A[ B[ C[ D]]]
BT <A B C>
BT <A B>[ C[ D]]
BT (A B)[ C[ D]]
BT (A B C)[ D]
A-D ARE ANY OF H3~6
>
```

2.2 Breakpoint Syntax

For H3/H4

addx - is an up to 4 digit hex address with an optional byte pair of "XX".
data - is an up to 2 digit hex data.
<dbwc> - is an up to 2 digit hex setting; interpreted as 16 binary data bits, where "0" indicates "don't care".
q - is a qualifier for the type of processor activity, where "X" indicates "don't care".
cnt - is a trigger count; the range is 1~4000H (for nth match).
dly - is a cycle count delay; the range is 2~FFFFH.

For a detailed description refer to section 2.3.

For H5/H6

H5 [**xx**] - are 2 spare bits (X1 and X0).
H6 [**xx**] - are 2 spare bits (X3 and X2).

Spare bits are monitored signals input from stick headers X0~3 on the CEP board. They can be set to match signal input at HIGH, LOW or "don't care" (1/0/X). For a detailed description refer to section 2.4.

2.3 Data Breakpoints (H3/H4)

H3 [addx[data [<dbwc>][qualifier[count]]]]
H4 [addx[data [<dbwc>][qualifier[delay]]]]

H3 is the command for a Data Breakpoint with an optional event count.

H4 is the command for a Data Breakpoint with an optional cycle delay.

addx is the trigger address where emulation is to stop. This can be an up to four digit hex address with an optional byte pair of "XX". (The maximum address length depends on the address range for the current emulation processor.)

data is an up to 2 digit hex data to be matched with the trigger address.

<dbwc> is an up to 2 digit hex setting indicating a "**data bit wildcard**". To enter any <dbwc>, a **data** value must first be defined. The data bit wildcard is interpreted as 16 binary data bits, where "0" indicates "don't care". The default value for the dbwc is FFH.

qualifier is a single or double alphabetic character indicating the type of processor activity to be matched with the trigger address. (If a **count** or **delay** is to be input, an "X" may be used in the status specification to indicate "don't care".)

For a complete listing of qualifiers applicable for a particular processor, refer to the appropriate appendix in the NEW MICE-II User's Manual.

count is a hexadecimal value from 1~4000H that specifies the number of times the target condition must be matched before the trace stops. To enter a count, a **qualifer** must first be defined.

delay is a hexadecimal value from 2~FFFFH that specifies a cycle count delay. After all other trigger conditions have been matched, the trace will continue until the specified cycle count has expired. To enter a delay, a **qualifier** must first be defined.

The breakpoints H3 and H4 are latched on the rising edge of a $\overline{\text{STROBE}}$ signal from the CEP board. When the breakpoint is activated, emulation stops at the end of the next bus cycle. (See Appendix C for a detailed description of break operation.) If a trigger count or cycle delay is defined, then the specified value must also be matched before the breakpoint is activated. This method is useful for tracing data after all other trigger conditions have been matched.

When the specified break condition is matched, a sync signal is output. Note that **delay** is not calculated for

the sync signal; as soon as all other trigger conditions are matched, the sync signal is transmitted without waiting for the specified delay to transpire. A count, however, specifies the number of times a sync signal will be transmitted. Everytime the trigger condition is matched and the count incremented, and a sync signal is output.

The following sample program for an Z80 target is used as the basis for all the examples listed in this user's manual. This program includes commonly used instructions for performing various bus cycles. Without a target system connected, it executes in a simple loop and select BUSAK mode by X13 in CEP board.

```
>J0
>Z
LOC     OBJ              LINE     LABEL     SOURCE CODE
0000    00               0001     B0000     NOP
0001    00               0002               NOP
0002    210003           0003               LD    HL,0300
0005    3E55             0004               LD    A,55
0007    77               0005               LD    (HL),A
0008    DB00             0006               IN    A,(00)
000A    00               0007               NOP
000B    D300             0008               OUT   (00),A
000D    110002           0009               LD    DE,0200
0010    12               0010     B0010     LD    (DE),A
0011    13               0011               INC   DE
0012    3C               0012               INC   A
0013    FE0F             0013               CP    0F
0015    20F9             0014               JR    NZ,0010
0017    C30000           0015               JP    0000
001A    00               0016               NOP
     DISASSEMBLY COMPLETED
>
```

Example 1: Use the "H?" command to display all breakpoint settings; then set H3 at address 0000H and begin program execution.

```
>H?                 ;Display H1-H6 settings.
H1=
H2=
H3=
H4=
H5=
H6=
>J0                 ;Reset the PC to 0000H.
>H3 0
>R                  ;Examine the registers prior to program execution.
  A  B  C  D  E  H  L  I  F    X    Y    S   PC
 FF BF FF 02 00 03 00 00 4A BFBF FFFF FFFF 0000
>G                  ;Breakpoint matched at 0000H.
PROGRAM BROKE AT BREAKPOINT-3
>C                  ;Cycle command steps the program to address 01H.
      IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
               0001   00    ?      11111111
        0002   0002   21    S      11111111
               0003   00    R      11111111 <ESC>
>
```

Note: For the Z80 if H3/H4 has been set and the trigger condition is matched at the current cycle, the CPU will stop at the next cycle (see Appendix C).

Example 2: Set H3 for any address with "don't care" and data for 0000. (Note that without a dbwc, all binary bits of the specified data value are qualified.)

```
>J0                          ;Reset the PC to 0000H.
>H3 XXXX 00                  ;Use H3 to break at data 00.
>G
PROGRAM BROKE AT BREAKPOINT-3
>C
      IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
               0001   00    ?      11111111
        0002   0002   21    S      11111111
               0003   00    R      11111111 <ESC>
>
```

Example 3: Set H3 with address and data of "don't care"; then set the status to Port Input with a count of 3.

```
>H0                          ;Clear H1-H6 settings.
>H3 XXXX 00 <0> I 3          ;H3 set to break on 3rd input cycle.
>J0                          ;Reset the PC to 0000H.
>G
PROGRAM BROKE AT BREAKPOINT-3
>C
      IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
               000B   D3    ?      11111101
               000C   00    F      11110000
               FF00   FF    C      11111111
        000D   000D   11    S      11110000 <ESC>
>
```

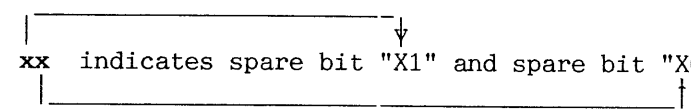
Example 4: Set H4 to any address XXXXH with a data bit wildcard that qualifies bit 1 only; then set the status to Port Output with a delay of 2.

```
>J0                      ;Reset the PC to 0000H.
>H0                      ;Clear H1-H6 settings.
>H4 XXXX FF <2> O 2      ;H4 set for address XXXXH, data "xxxx xx1x".
>H?                      ;Display H1-H6 settings.
H1=
H2=
H3=
H4= XXXX FF <2> O 2
H5=
H6=
>G
PROGRAM BROKE AT BREAKPOINT-4
>C
       IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
              0012    3C     ?      11110010
         0013 0013    FE     S      11111111
              0014    0F     R      11110100 <ESC>
>
```

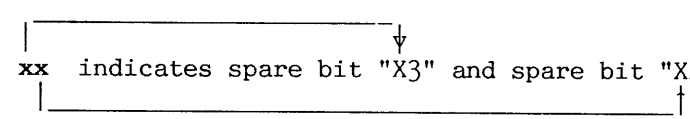
2.4 External Hardware Breakpoints (H5/H6)

H5 [xx]
H6 [xx]

H5 is the command for an External Hardware Breakpoint set by monitored signals at X1 and X0 on the CEP board.



H6 is the command for an External Hardware Breakpoint set by monitored signals at X3 and X2 on the CEP board.



The breakpoints H5 and H6 use two external signal groups on the CEP board (X1/X0 and X3/X2 respectively). Mini-jumpers provided with the BPP can be used to connect from these test points to the target. The monitored signals are called spare bits and support concurrent trace for address, data and status bus. Any of these bits can be monitored to provide hardware status based on machine cycles. Hardware breakpoints are latched when the specified logic state occurs.

The spare bits input from stick headers X0~3 on the CEP board can be set to match signal input at HIGH, LOW or "don't care" (1/0/X), where

1 = high level
0 = low level
X = don't care (either 1 or 0)

Note: Input signals to X0~3 must meet specifications for the Schmitt trigger IC - 74LS132.

To demonstrate External Hardward Breakpoints in the following examples, jumpers are connected to the CEP board of a NEW MICE-II Z80. The spare bits X0, X1, X2, X3 are configured as shown below:

X0 is connected to U56-18 (CEP 74F244 $\overline{\text{IORQ\_T}}$)
X1 is connected to U56-16 (CEP 74F244 $\overline{\text{M1\_T}}$)
X2 is connected to U56-12 (CEP 74F244 $\overline{\text{RD\_T}}$)
X3 is connected to U56-3 (CEP 74F244 $\overline{\text{WR\_T}}$)

Example 1: Use H5 to break emulation after an I/O cycle by setting the input trigger at 10 for X1/X0.

```
>H0                          ;Clear H1-H6 settings.
>J0                          ;Reset the PC to 0000H.
>H5 10                       ;Break condition for X0=0, X1=1.
>H?                          ;Display H1-H6 settings.
H1=
H2=
H3=
H4=
H5= 10
H6=
>G
PROGRAM BROKE AT BREAKPOINT-5
>C
       IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
                000A   00    ?      11111001
         000B   000B   D3    S      11111001
                000C   00    P      11111011 <ESC>
>
```

Example 2: Use H6 to break emulation after an Write cycle by setting the input trigger at 01 for X3/X2.

```
>J0                          ;Reset the PC to 0000H.
>H0                          ;Clear H1-H6 settings.
>H6 01                       ;Set break condition for X2=1, X3=0.
>H?                          ;Display H1-H6 settings.
H1=
H2=
H3=
H4=
H5=
H6= 01
>G
PROGRAM BROKE AT BREAKPOINT-6
>C
      IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
               000D   11    ?      11111001
               000E   00    R      11111011
               000F   02    R      11111011 <ESC>
>
```

SECTION 3

BREAKPOINT TRACE LOGIC

3.1 Breakpoint Trace - BT

Refer to the following sections for description of syntax.

BT is the command for Breakpoint Tracing.

Breakpoint Tracing begins at the current address (Backward Trace) and continues until all the specified breakpoints are matched. H3~H6 must be set prior to command execution, and may be specified as a single breakpoint or in logical combinations (ARM/AND/OR) with the BT command. Note that H1 and H2 (specified prior to execution, and not within the BT command definition) can also be used to add another logical OR to the trigger construct (refer to the last example in section 3.1.1). The BPP performs the following types of tracing:

Event Count Trace
Delay Trigger Trace
ARM Trigger Trace
AND Trigger Trace
OR Trigger Trace

3.1.1 Event Count Trace (with H3)

The address, data and qualifier of a specific program event can be defined along with a **count** in the H3 breakpoint. When the Event Count trigger is enabled, it is added as a logical OR to the breakpoint parameters defined in B/F commands. When enabled and then defined in the BT command, it can be used alone or in any legal combination with other breakpoints. (Also refer to the example in section 3.1.3.)

Example: Set break conditions for H3 at any address, any data, Write cycle and with a count of 2; then execute a trace using the BT command.

```
>J0                              ;Reset the PC to 0000H.
>H0                              ;Clear H1-H6 settings.
>H3 XXXX FF <0> W 2              ;Set the event count trigger.
>H?                              ;Display H1-H6 settings.
H1=
H2=
H3= XXXX FF <0> W 2
H4=
H5=
H6=
>BTH3                            ;Execute a Breakpoint Trace with H3.
THE TRACE STOPS AT STEP 0015
>L0 0 FFFF W
 FRAME IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
  0008          0300   55    W      11111000
  0014          0200   FF    W      11111111
>
```

Example: Set the break condition for H1 at address 0H, then execute a trace again using the BT command.

```
>J0                                ;Reset the IP to 0000H.
>H1 0                              ;Set H1 to address 0000H.
>BTH3
THE TRACE STOPS AT STEP 0000
PROGRAM BROKE AT BREAKPOINT-1      ;Tracing is stopped by H1.
>
```

3.1.2 Delay Trigger Trace (with H4)

The address, data and qualifier of specific program activity can be defined along with a **delay** in the H4 breakpoint. When the Delay Trigger is enabled, it is added as a logical OR to the breakpoint parameters defined in the B/F commands. When enabled and then defined in the BT command, it can be used alone or in any legal combination with other breakpoints. (Also refer the example in section 3.1.3.)

Example: Set the break conditions for H4 at address 200H, data FFH, Write cycle and with a delay of 5 machine cycles; then execute tracing using the BT command.

```
>J0                                ;Reset the PC to 0000H.
>H0                                ;Clear H1-H6 settings.
>H4 200 FF W 5                     ;H4 set address for 200, data FFH,
                                   ; write status with delay 5 cycles.
>H?                                ;Display H1-H6 settings.
H1=
H2=
H3=
```

```
H4= 200 FF W 5
H5=
H6=
>BTH4                          ;Execute trace.
THE TRACE STOPS AT STEP 0019
>L                             ;List all traced data.
 FRAME IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
  0000         0000   00   ?      10111110
  0001   0001   0001   00   S      11110000
  0002   0002   0002   21   S      11110010
  0003         0003   00   R      11110000
  0004         0004   03   R      11110100
  0005   0005   0005   3E   S      11110110
  0006         0006   55   R      11111000
  0007   0007   0007   77   S      11111110
  0008         0300   55   W      11111000
  0009   0008   0008   DB   S      11111101
  000A         0009   00   R      11110000
  000B         5500   FF   I      11111111
  000C   000A   000A   00   S      11110000
  000D   000B   000B   D3   S      11111101
  000E         000C   00   R      11110000
  000F         FF00   FF   O      11111111
  0010   000D   000D   11   S      11110000
  0011         000E   00   R      11110000
  0012         000F   02   R      11110100
  0013   0010   0010   12   S      11110100
  0014         0200   FF   W      11111111   ;H4 setting matched.
  0015   0011   0011   13   S      11110100-
  0016   0012   0012   3C   S      11110010 |
  0017   0013   0013   FE   S      11111111 |->;Trace delayed for 5
  0018         0014   0F   R      11110100 |  ; cycles then stopped.
  0019   0015   0015   20   S      11110010-|
>
```

3.1.3 Breakpoint Interval Timer

A timer is also supported for trigger trace mode which displays the interval from the initial trigger until emulation stops. (The interval will only display when an Armed breakpoint is encountered.) The timer can run for up to 35.39 minutes before the counter is reset. Note that resolution for the interval is down to micro-seconds. Time interval message syntax is as follows:

INITIAL TRIGGER TO BREAKPOINT INTERVAL-- aa:bb:ccc:ddd

where: **aa** = minute
bb = second
ccc = millisecond
ddd = microsecond

Example: Display the execution interval for an Armed Trigger Trace. (Refer to section 3.2 for a description of the Arm Trigger.)

```
>J0                              ;Reset the PC to 0000H.
>H3 XXXX FF 0
>H?                              ;Display H1-H6 settings.
H1=
H2=
H3= XXXX FF 0
H4= 200 FF W 5
H5=
H6=
>BTH3 H4                         ;Execute BT command with H3, H4.
THE TRACE STOPS AT STEP 0019
INITIAL TRIGGER TO BREAKPOINT INTERVAL--00:00:000:005
```

```
>L
 FRAME IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
  0000         B7D7   7E    ?      10111110
  0001   0001  0001   00    S      11110000
  0002   0002  0002   21    S      11110010
  0003         0003   00    R      11110000
  0004         0004   03    R      11110100
  0005   0005  0005   3E    S      11110110
  0006         0006   55    R      11111000
  0007   0007  0007   77    S      11111110
  0008         0300   55    W      11111000
  0009   0008  0008   DB    S      11111101
  000A         0009   00    R      11110000
  000B         5500   FF    I      11111111
  000C   000A  000A   00    S      11110000
  000D   000B  000B   D3    S      11111101
  000E         000C   00    R      11110000
  000F         FF00   FF    O      11111111   ;H3 matched.
  0010   000D  000D   11    S      11110000
  0011         000E   00    R      11110000
  0012         000F   02    R      11110100
  0013   0010  0010   12    S      11110100
  0014         0200   FF    W      11111111   ;H4 setting matched.
  0015   0011  0011   13    S      11110100-
  0016   0012  0012   3C    S      11110010 |
  0017   0013  0013   FE    S      11111111 |->;Emulation stops
  0018         0014   0F    R      11110100 |  ; after 5 cycles.
  0019   0015  0015   20    S      11110010-|
>
```

3.2 Armed Trigger Trace - BT

BT A[B[C[D]]]

A[B[C[D]]] Breakpoints A-D (which represent any of H3~6) can be logically joined in any Armed Trigger sequence. (The only exception being that they cannot be repeated).

There are up to 64 different Armed Trigger combinations. Sequential tracing is executed as follows -

if trigger A is matched, then trigger B is enabled;
if trigger B is matched, then trigger C is enabled;
and so on until the final breakpoint is matched.

Example 1: Execute a trigger trace using the single breakpoint of H5 set at (X1=1,X0=0).

```
>J0                              ;Reset the PC to 0000H.
>H5 10                           ;Break condition for X0=0, X1=1.
>H?                              ;Display H1-H6 settings.
H1=
H2=
H3= XXXX FF 0
H4= 200 FF W 5
H5= 10
H6=
>BTH5                            ;Execute trace.
THE TRACE STOPS AT STEP 000B
>L                               ;List all trace data.
 FRAME IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
  0000          0000  00   ?      10111111
  0001   0001    0001  00   S      11111111
  0002   0002    0002  21   S      11111111
  0003          0003  00   R      11111111
  0004          0004  03   R      11111111
  0005   0005    0005  3E   S      11111111
  0006          0006  55   R      11111111
  0007   0007    0007  77   S      11111111
  0008          0300  55   W      11111111
  0009   0008    0008  DB   S      11111111
  000A          0009  00   R      11111111
  000B          5500  FF   I      11111111  ;H5 matched.
>
```

Example 2: Execute an armed trigger trace with a single sequence of H5-->H6.

```
>J0                              ;Reset the PC to 0000H.
>H?                              ;Display H1-H6 settings.
H1=
H2=
H3= XXXX FF 0
H4= 200 FF W 5
H5= 10                           ;Break condition for X0=0, X1=1.
H6= 01                           ;Break condition for X2=1, X3=0.
>BTH5 H6
THE TRACE STOPS AT STEP 000F
INITIAL TRIGGER TO BREAKPOINT INTERVAL--00:00:000:002
>L                               ;List all traced data.
 FRAME IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
  0000         0000   00   ?      11111111
  0001   0001   0001   00   S      11111111
  0002   0002   0002   21   S      11111111
  0003         0003   00   R      11111111
  0004         0004   03   R      11111111
  0005   0005   0005   3E   S      11111111
  0006         0006   55   R      11111111
  0007   0007   0007   77   S      11111111
  0008         0300   55   W      11111111
  0009   0008   0008   DB   S      11111111  ;H5 matched.
  000A         0009   00   R      11111111
  000B         5500   FF   I      11111111
  000C   000A   000A   00   S      11111111
  000D   000B   000B   D3   S      11111111
  000E         000C   00   R      11111111
  000F         FF00   FF   O      11111111  ;H6 matched.
  >
```

Example 3: Execute an armed trigger trace using a sequence of twc triggers and a final breakpoint, i.e. H6-->H5-->H3.

```
>J0                          ;Reset the PC to 0000H.
>H6 01                       ;Break condition for X2=1, X3=0.
>H?                          ;Display H1-H6 settings.
H1=
H2=
H3= XXXX FF 0
H4= 200 FF W 5
H5= 10
H6= 01
>BTH6 H5 H3                  ;Execute the logical trace H6→H5→H3.
THE TRACE STOPS AT STEP 0010
INITIAL TRIGGER TO BREAKPOINT INTERVAL--00:00:000:003
>L
 FRAME IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
  0000          0000   00    ?      10111110
  0001   0001   0001   00    S      11110000
  0002   0002   0002   21    S      11110010
  0003          0003   00    R      11110000
  0004          0004   03    R      11110100
  0005   0005   0005   3E    S      11110110
  0006          0006   55    R      11111000
  0007   0007   0007   77    S      11111110
  0008          0300   55    W      11111000  ;H6 matched.
  0009   0008   0008   DB    S      11111101
  000A          0009   00    R      11110000
  000B          5500   FF    I      11111111  ;H5 matched.
  000C   000A   000A   00    S      11110000
  000D   000B   000B   D3    S      11111101
  000E          000C   00    R      11110000
  000F          FF00   FF    O      11111111  ;H3 matched.
  0010   000D   000D   11    S      11110000
>
```

Example 4: Execute an armed trigger trace using a sequence of three triggers and a final breakpoint, i.e. H5-->H6-->H3-->H4.

```
>J0                                  ;Reset the PC to 0000H.
>H5 10                               ;Break condition for X0=0, X1=1.
>H?                                  ;Display H1-H6 settings.
H1=
H2=
H3= XXXX FF 0
H4= 200 FF W 5
H5= 10
H6= 01
>BTH5 H6 H3 H4                       ;Execute the logical trace.
THE TRACE STOPS AT STEP 001A
INITIAL TRIGGER TO BREAKPOINT INTERVAL--00:00:000:013
>L0 0 FFFF I O W                     ;List all INPUT/OUTPUT/WRITE cycles.
 FRAME IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
  0008          0300   55   W      11111000
  000B          5500   FF   I      11111111
  000F          FF00   FF   O      11111111
  0014          0200   FF   W      11111111
>
```

3.3 AND Trigger Trace - BT

BT(A B C)[D]
BT(A B)[C[D]]

(A B C)[D] The AND trigger (breakpoints A, B and C) can be used as a single breakpoint construct, or can serve as a trigger construct for breakpoint D.

(A B)[C [D]] The AND trigger (breakpoints A and B) can be used as a single breakpoint construct, or can serve as a trigger construct for breakpoint C or for the armed trigger construct of C-->D.

Breakpoints A-D can be combined in a two or three level AND construct. Armed breakpoints can also be specified following the AND trigger. There are up to 38 different AND trigger combinations. Note that parantheses "( )" must be used in the syntax when inputting an AND trigger construct.

Example 1: Run a trace with an AND trigger used as a single breakpoint construct with H4, H5 and H6. (Note that breakpoints specified in an AND construct can be matched in any order.)

```
>J0                                ;Reset the PC to 0000H.
>H0                                ;Clear H1-H6 settings.
>H3 XXXX FF <0> W 2
>H4 200 FF W 5
>H5 10
>H6 01
>H?                                ;Display H1-H6 settings.
H1=
H2=
H3= XXXX FF <0> W 2
H4= 200 FF W 5
H5= 10
H6= 01
>BT(H5 H6 H4)                      ;Trace with AND trigger.
THE TRACE STOPS AT STEP 001B
>L                                 ;List all traced data.
 FRAME IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
  0000        0000    00   ?      10111110
  0001  0001  0001    00   S      11110000
  0002  0002  0002    21   S      11110010
  0003        0003    00   R      11110000
  0004        0004    03   R      11110100
  0005  0005  0005    3E   S      11110110
  0006        0006    55   R      11111000
  0007  0007  0007    77   S      11111110
  0008        0300    55   W      11111000
  0009  0008  0008    DB   S      11111101  ;H5 matched.
  000A        0009    00   R      11110000
  000B        5500    FF   I      11111111
  000C  000A  000A    00   S      11110000
```

```
000D    000B    000B    D3    S     11111101
000E            000C    00    R     11110000
000F            FF00    FF    O     11111111
0010    000D    000D    11    S     11110000
0011            000E    00    R     11110000
0012            000F    02    R     11110100
0013    0010    0010    12    S     11110100  ;H6 matched.
0014            0200    FF    W     11111111
0015    0011    0011    13    S     11110100
0016    0012    0012    3C    S     11110010
0017    0013    0013    FE    S     11111111
0018            0014    0F    R     11110100
0019    0015    0015    20    S     11110010
001A            0016    F9    R     11111111  ;H4 matched.*
001B    0010    0010    12    S     11111111
>
```

* H5, H6 and H4 all matched; breakpoint H4 stops emulation at the next cycle.

Example 2: Perform a trace using an AND construct as the trigger for an armed breakpoint.

```
>J0                              ;Reseta the PC to 0000H.
>H?                              ;Display H1-H6 settings.
H1=
H2=
H3= XXXX FF <0> W 2
H4= 200 FF W 5
H5= 10
H6= 01
>BT(H4 H5 H6) H3                 ;Execute the logical trace.
THE TRACE STOPS AT STEP 0025
INITIAL TRIGGER TO BREAKPOINT INTERVAL--00:00:000:022
>L0 0 FFFF I O W                 ;List all INPUT/OUTPUT/WRITE cycles
                                 ; between 0H-FFFFH, start at frame 0.
 FRAME IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
  0008         0300   55    W      11111000
  000B         5500   FF    I      11111111
  000F         FF00   FF    O      11111111
  0014         0200   FF    W      11111111
  001C         0201   00    W      11110000
  0024         0202   01    W      11110000
>
```

Example 3: Run a trace with an AND trigg and an armed breakpoint sequence.

```
>J0                          ;Reset the PC to 0000H.
>H?                          ;Display H1-H6 settings.
H1=
H2=
H3= XXXX FF <0> W 2
H4= 200 FF W 5
H5= 10
H6= 01
>BT(H6 H5) H4 H3             ;Execute an AND trigger and H4→H3
                             ; as an armed breakpoint.
THE TRACE STOPS AT STEP 0025
INITIAL TRIGGER TO BREAKPOINT INTERVAL--00:00:000:021
>LO 0 FFFF I O W             ;List all INPUT/OUTPUT/WRITE cycles
                             ; between 0H-FFFFH, start frame 0.
 FRAME IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
  0008          0300   55    W      11111000
  000B          5500   FF    I      11111111
  000F          FF00   FF    O      11111111
  0014          0200   FF    W      11111111  ;H4, H3 matched.
  001C          0201   00    W      11110000
  0024          0202   01    W      11110000
>
```

3.4 OR Trigger Trace - BT

BT<A B C>
BT<A B>[C[D]]

<A B C> The OR trigger (breakpoints A, B or C) can be used as a single breakpoint construct.

<A B>[C [D]] The OR trigger (breakpoints A or B) can be used a single breakpoint construct, or can serve as a trigger for breakpoint C or for the armed trigger construct C-->D.

Breakpoints A-D can be combined in a two or three level OR construct. Armed breakpoints can also be specified after the OR trigger. If an armed trigger is used with the OR construct, the ORed pairs must be either data (H3,H4) or external breakpoints (H5,H6). There are up to 18 different OR trigger combinations. Note that angular brackets "< >" must be used in the syntax when inputting an OR trigger construct.

Example 1: Execute a trace using a two level OR construct.

```
>J0                          ;Reset the PC to 0000H.
>H?                          ;Display H1-H6 settings.
H1=
H2=
H3= XXXX FF <0> W 2
H4= 200 FF W 5
H5= 10
H6= 01
```

```
>BT<H4 H5>                          ;Execute a trace with data breakpoint pair.
THE TRACE STOPS AT STEP 000B
>L                                  ;List all traced data.
 FRAME IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
  0000         0000    00    ?      10111110
  0001   0001   0001    00    S      11110000
  0002   0002   0002    21    S      11110010
  0003         0003    00    R      11110000
  0004         0004    03    R      11110100
  0005   0005  0005    3E    S      11110110
  0006         0006    55    R      11111000
  0007   0007  0007    77    S      11111110
  0008         0300    55    W      11111000
  0009   0008  0008    DB    S      11111101
  000A         0009    00    R      11111111  ;H5 matched.
  000B         5500    FF    I      11111111
>
```

Example 2: Execute a trace using an OR construct as the trigger for an armed breakpoint.

```
>J0                                 ;Reset the PC to 0000H.
>H3 XXXX FF <0> W
>H?                                 ;Display H1-H6 settings.
H1=
H2=
H3= XXXX FF <0> W
H4= 200 FF W 5
H5= 10
H6= 01
>BT<H5 H6> H3                       ;Execute the trace.
THE TRACE STOPS AT STEP 001D
INITIAL TRIGGER TO BREAKPOINT INTERVAL--00:00:000:009
```

```
>L                                  ;List all trace data.
 FRAME IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
  0000         0000   00    ?      10111110
  0001   0001   0001   00    S      11110000
  0002   0002   0002   21    S      11110010
  0003         0003   00    R      11110000
  0004         0004   03    R      11110100
  0005   0005   0005   3E    S      11110110
  0006         0006   55    R      11111000
  0007   0007   0007   77    S      11111110
  0008         0300   55    W      11111000
  0009   0008   0008   DB    S      11111101
  000A         0009   00    R      11110000
  000B         5500   FF    I      11111111  ;H5 matched.
  000C   000A   000A   00    S      11110000
  000D   000B   000B   D3    S      11111101
  000E         000C   00    R      11110000
  000F         FF00   FF    O      11111111
  0010   000D   000D   11    S      11110000
  0011         000E   00    R      11110000
  0012         000F   02    R      11110100
  0013   0010   0010   12    S      11110100
  0014         0200   FF    W      11111111  ;H3 matched.
  0015   0011   0011   13    S      11110100
>
```

Example 3: Run a trace using an OR trigger and an armed breakpoint sequence.

```
>J0                              ;Reset the PC to 0000H.
>H5 10                           ;Break condition or X0=0, X1=1.
>H3 XXXX FF <0> W
>H?
H1=
H2=
H3= XXXX FF <0> W
H4= 200 FF W 5
H5= 10
H6= 01
>BT<H3 H4> H5 H6                 ;Execute an OR trigger and an armed
                                 ; breakpoint of H5→H6.
THE TRACE STOPS AT STEP 000F
INITIAL TRIGGER TO BREAKPOINT INTERVAL--00:00:000:003
>L0 0 FFFF I O W
 FRAME IFADDR ADDRESS DATA STATUS SPARE(8 BITS)
  0008          0300   55    W      11111000  ;H3 matched.
  000B          5500   FF    I      11111111
  000F          FF00   FF    O      11111111
>
```

SECTION 4

SYNC SIGNAL OUTPUT

4.1 External Hardware Trigger Output - BS

BS 1|2[addx[data[<dbwc>][qualifier]]]

BS — is the command for External Hardware Trigger Output.

1 — specifies a positive signal pulse of 140~180ns. (Executing BS1 clears H3 because the same hardware circuit is used to support both of these breakpoints.)

2 — specifies a positive signal pulse that lasts for the duration of the trigger condition. (Executing BS2 clears H4 because the same hardware circuit is used to support both of these breakpoints.)

addx — is the trigger address that causes output of a sync signal. This can be an up to four digit hex address with an optional byte pair of "XX". (The maximum address length depends on the address range of the current emulation processor.)

data — is an up to 2 digit hex data to be matched with the trigger address.

<dbwc> is an up to 2 digit hex setting indicating a "data bit wildcard". To enter any <dbwc>, a data value must first be defined. The data bit wildcard is interpreted as 8 binary data bits, where "0" indicates "don't care". The default value for the dbwc is FFH.

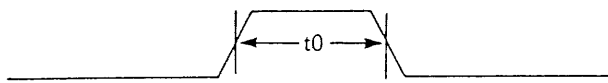
qualifier is a single or double alphabetic character indicating the type of processor activity to be matched with the trigger address. For a complete listing of the qualifiers applicable for a particular processor, refer to the appendices in the NEW MICE-II User's Manual.

The BS command can be used to trigger a sync signal to an external device (e.g. oscilloscope or logic analyzer). Connection is made on the BPP board at BSYNC1 (J3-1) or BSYNC2 (J3-2).

When the specified parameters are matched, a sync signal is transmitted. If no other breakpoints are specified before the BS command, emulation will continue after the trigger condition is met. To stop emulation before execution has been completed, input a Halt command.

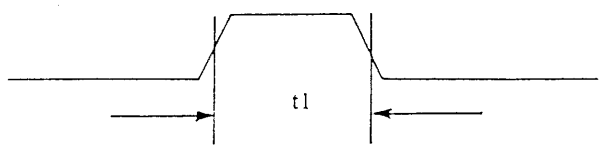
4.2 Output Signal Timing

1) BSYNC 1 Output Timing



When the BS1 setting is matched, the BPP will issue a positive signal pulse as indicated in the figure above. The pulse width of t0 = 140~180ns.

2) BSYNC 2 Output Timing



When the BS2 setting is matched, the BPP will issue a positive signal pulse as indicated in the figure above. The pulse width of t1 is determined by the following conditions:

a) No "wait state" inserted

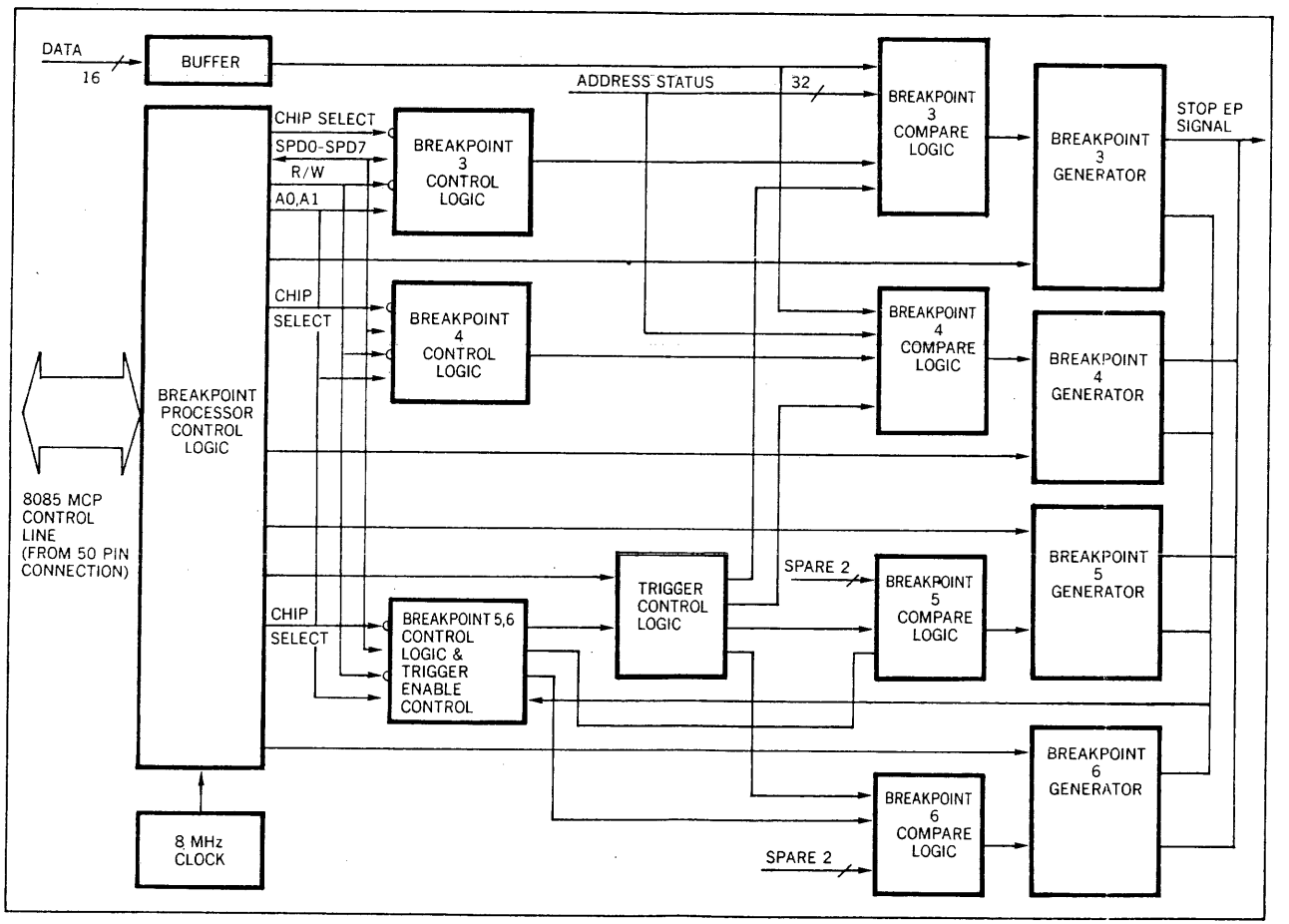
Pulse width of t1 = duration the trigger condition is matched on the bus (which depends on bus timing for the particular target system).

b) "Wait state" inserted

Pulse width of t1 = duration the trigger condition is valid on the bus, plus timing for wait state.

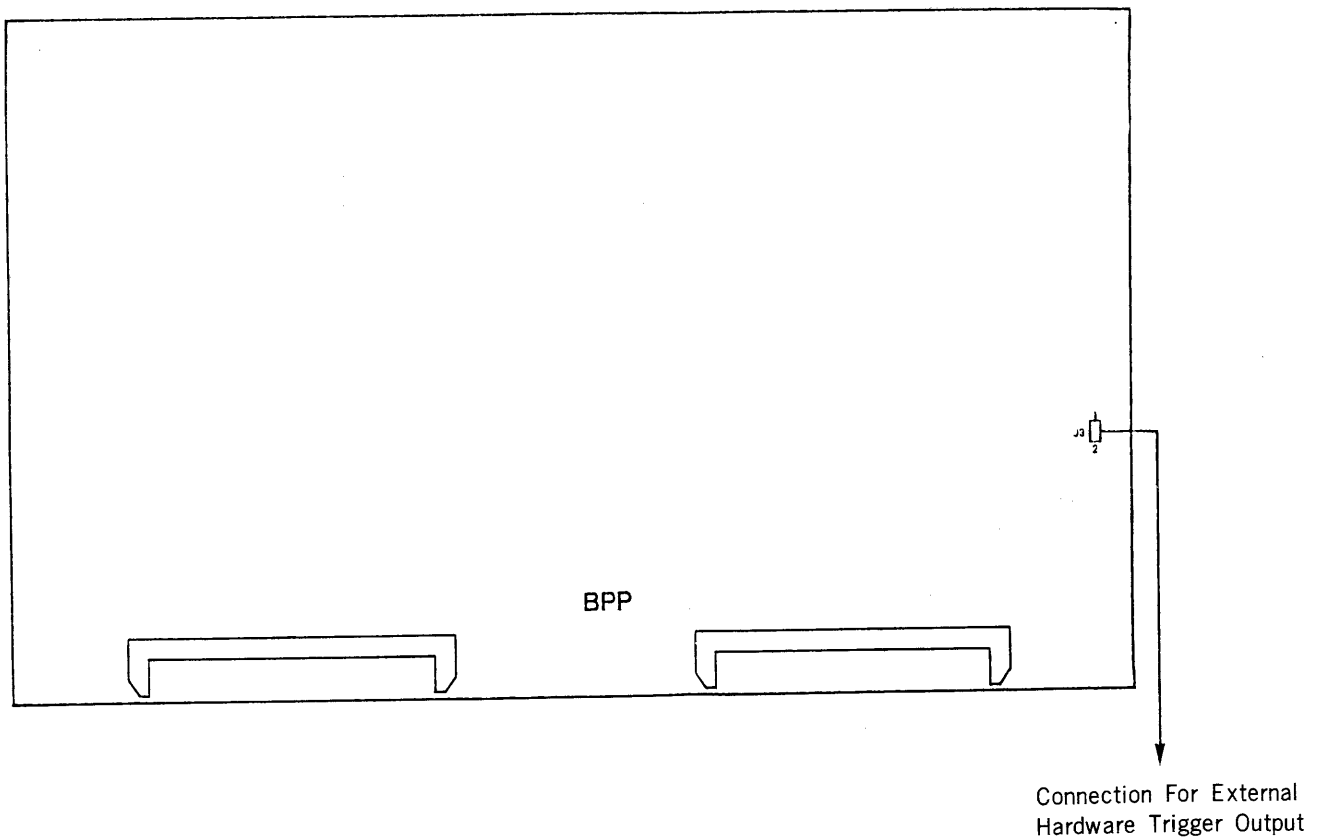
APPENDIX A

BPP BLOCK DIAGRAM



APPENDIX B

BPP PLACEMENT CHART



APPENDIX C

TIMING FOR BREAK OPERATION (H1-H6)

C.1 Emulation Status

NEW MICE-II performs emulation in two different ways, depending on the type of processor.

1) **Hold State**

   The emulation processor bus is at high impedence whenever it stops. In this state all signals on the bus are floating and cannot be tested. Hold state is used in NEW MICE-II Z80.

2) **Wait State**

   The emulation processor bus is in wait state whenever it stops. In this state all signals on the bus are stable and can be tested with a logic probe or oscilloscope. Wait state is used in NEW MICE-II NSC800. In wait state, the bus indicates the current address for the NSC800.

3) Run State

Whenever user's program execution is stopped, the emulation processor is in free-run state, executing "JR $" instructions. In this state the signals at the Input/Output pins are stable and can be tested, but the signals on the Address/Data bus are changing dynamically and cannot be tested. Run state is used for NEW MICE-II Z8 and ZS8.

The type of emulation performed determines timing for breakpoint operation. The following sections describe break operation for H1-H6. (The examples for hold state are based on the Z80, examples for wait state are based on the NSC800 and those for run state are based on the Z8.)

C.2 Timing For Breakpoints H1/H2

1) Hold State

Emulation stops at the end of the bus cycle* in which the break condition is matched; thebus cycle is completed.

*bus cycle: complete operation of bus activity such as read/write data from/to memory, or input/output data from/to I/O port.

Now input a Cycle Step command to display the address of the next cycle (i.e. PC+1H for 8-bit systems,assuming no branch instruction is executed). Entering an Register command instead of the C command at this time may show that the PC does not equal PC+1H, though. If the current instruction has not yet been completed, the R command will finish it. Therefore, the PC displayed in the R command always points to the starting address of next unexecuted instruction; as illustrated by the program segment below.

```
ADDRESS   DATA     STATUS   SOURCE CODE
  :         :        :          :
0008        00       S        NOP
0009        00       S        NOP            ;*
000A        210003   S        LD HL,0300
000D                 S        NOP
  :         :        :          :
```

* The break condition (H1 or H2) is matched and emulation stops.

In this example, the break condition is matched and emulation stops at the end of the current bus cycle (address 0009H). The C command displays address 000AH, but an R command displays 000DH instead. Because the first byte (21H) of the instruction (LD HL,0300) was already fetched when emulation stopped, entering an R command completes execution of this instruction and displays PC=000DH. (See figure C-1, section C.5.)

2) **Wait State**

Emulation stops at stable bus state, before the current cycle has been completed. (The bus can be tested with a oscilloscope or logic probe at this time.) Entering a C command displays the current cycle address; but inputting an R command instead will display a different address if the status of the break cycle is other than S (instruction fetch). This is illustrated in the following program segment.

```
ADDRESS   DATA   STATUS   SOURCE CODE
   :        :       :          :
  0008      00      S       NOP
  0009      00      S       NOP
  000A      21      S       LD HL,0300    ;Break condition 1
  000B      00      R                     ;Break condition 2
  000C      03      R
  000D      00      S       NOP
   :        :       :          :
```

Note that no branch instruction is executed in the preceding example. In break condition 1, emulation is stopped at an instruction fetch cycle. Entering a C displays address 000AH. In break condition 2, however, emulation is stopped at a cycle where bus status is other than instruction fetch. Entering a C command displays 000BH; but inputting an R command finishes the instruction and displays PC=000DH. (See figure C-2 under section C.5.)

3) Run State

When the break condition is matched, emulation stops at the end of the current instruction. Entering a C command will execute the next instruction and then display the current status. However, entering an R command will display the current status without causing any instructions to be executed. This is illustrated in the following example.

```
ADDRESS   DATA     STATUS   SOURCE CODE
   :        :        :          :
 0000      FF       S        NOP
 0001      FF       S        NOP
 0002      561D41   S        AND    1D,#41     ;*
 0003      1D       R                           ;*
 0004      41       R                           ;*
 0005      8C       S        LD     R08,#44
 0006      44       R
 0007      FF       S        NOP
   :        :        :          :
```

* The specified break condition (H1 or H2) may be matched at any cycle of this instruction.

Entering a C command will execute the next instruction at address 0005H-0006H and then display the current status; but entering the R command instead will simply display the current status, where the PC=0005H. (See figure C-3, section C.5.)

C.3 Timing For Breakpoints H3/H4

The comparison operation for data breakpoints takes too long for emulation to be stopped at the current cycle. Also note that executing the C or R command after emulation has been stopped by H3 or H4 produces the same results as described above for H1/H2.

1) Hold State

When the break condition is matched, emulation stops at the end of the next bus cycle. (See figure C-1, section C.5.)

2) Wait State

When the break condition is matched, emulation stops at stable bus state for the next cycle. (See figure C-2, section C.5.)

3) Run State

When the break condition is matched, emulation is stopped at the end of the current instruction. However, if it is matched at the last cycle of an instruction; emulation will be stopped at the end of the next instruction. (See figure C-4, section C.5.)

C.4 Timing For Breakpoints H5/H6

1) Hold State

When H5 or H6 is matched, a signal is immediately (appx. 100ns) generated by MICE to stop emulation. Therefore, if the trigger occurs "early enough" in a bus cycle (e.g. before the last state of any M for Z80), emulation will stop at the end of the current cycle (same as H1/H2); but if the trigger occurs "too late" (e.g. after T2 for Z80), then emulation will be stopped at the end of the next cycle (same as H3/H4).

2) Wait State

When H5 or H6 is matched, a signal is immediately (appx. 100ns) generated by MICE to stop emulation at stable bus state. Therefore, if the trigger occurs "early enough" in a bus cycle (e.g. before T2 for NSC800), emulation will stop at valid state for the current cycle; but if the trigger occurs "too late" (e.g. after T2 for NSC800), then emulation will stop at valid state for the next cycle.

3) **Run State**

When H5 or H6 is matched, a signal is immediately (appx. 100ns) generated by MICE to stop emulation. Therefore, if the trigger occurs "early enough" (i.e. anywhere before the rising edge of $\overline{DS}$ or $\overline{MDS}$* for Z8/ZS8, during the last cycle of an instruction), emulation will be stopped at the end of the current instruction; but if the trigger occurs "too late" (i.e. after the rising edge of $\overline{DS}$ or $\overline{MDS}$ for Z8/ZS8, during the last cycle of an instruction), then emulation will be stopped at the end of the next instruction.

* $\overline{MDS}$ indicates internal program memory data strobe. This signal is output from the emulation CPU.

The break position for the emulation states described above is determined by the timing of H5/H6 and by the specific point in cycle execution that can be defined as "early enough" or "too late" (which depends on the type of emulation processor used, refer to section C.6).

C.5 Timing Diagrams For Break Operation

The following diagrams show the timing of break condition match and emulation halt during bus cycle execution.

1) Hold State

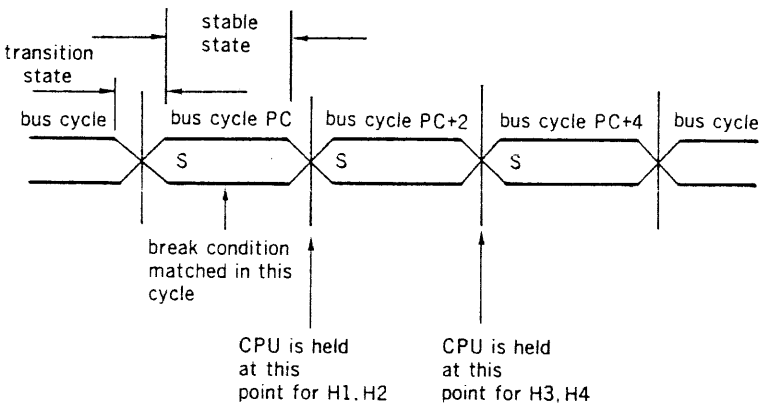


Figure C-1 Hold State Timing Diagram

S — is fetch state. (No I/O or R/W operation, or branch instruction is assumed in this example.)

PC — is the program counter. (The increment per bus cycle is PC=PC+1 for 8-bit microprocessors.

stable state — is a full bus cycle.

transition state — is the end of a bus cycle and the start of the next one.

2) Wait State

There are 4 bus cycles illustrated in the following diagram. (Note that this diagram is applicable for the Wait State example under section C.2.)

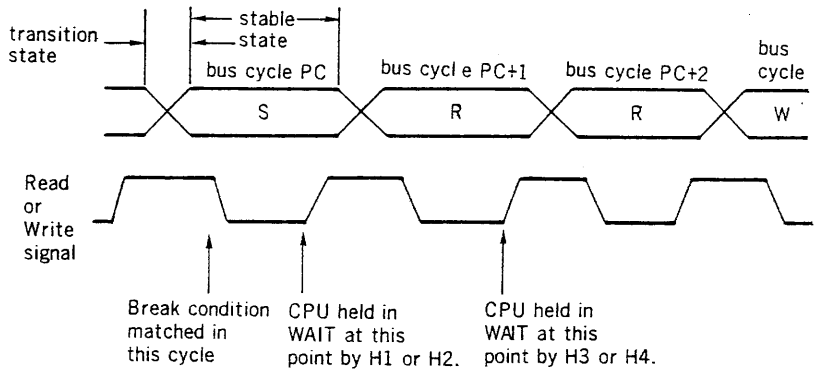


Figure C-2 Wait State Timing Diagram

S is an opcode fetch cycle (i.e. the 1st byte of an instruction; no branch instruction is assumed in this example).

R is a memory read cycle or the subsequent byte read for an instruction.

W is a memory write cycle.

PC	is the program counter. (The increment per bus cycle is PC=PC+1; the PC is 0008H in this example.)
stable state	is a full bus cycle.
transition state	is end of a bus cycle and start of the next one.

3) Run State

a) H1/H2

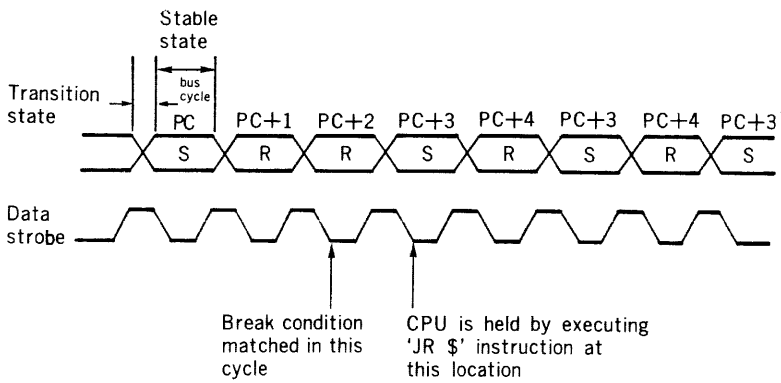


Figure C-3 Run State Timing Diagram for H1/H2

b) H3/H4

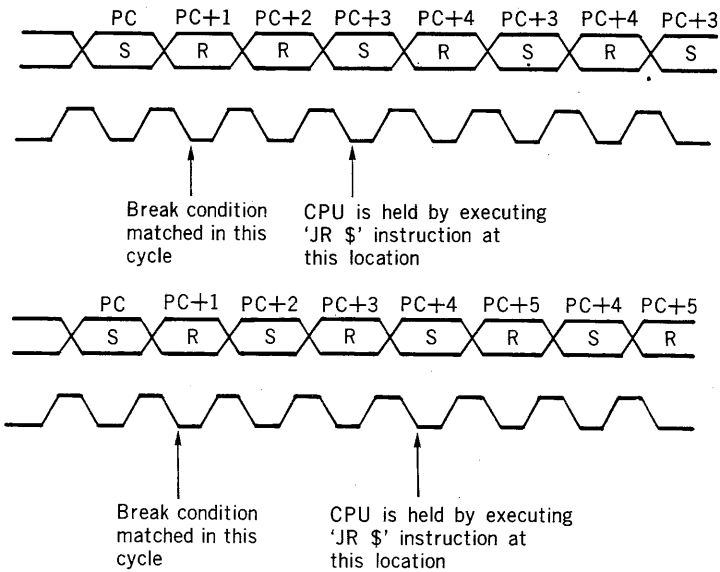


Figure C-4 Run State Timing Diagram for H3/H4

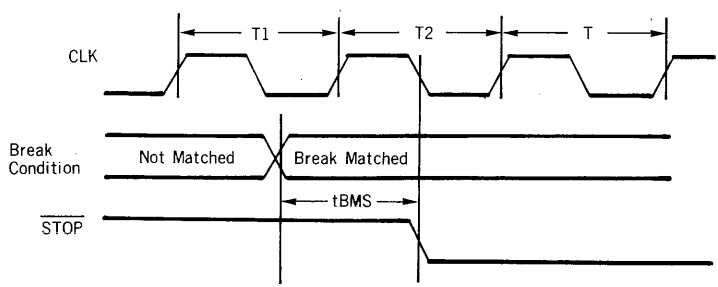
If execution is stopped by H3/H4, then the program counter will equal the halt address plus one instruction. However, if the halt address is at the last cycle of an instruction, the program counter will equal the halt address plus two instructions.

S	is an opcode fetch cycle (i.e. 1st byte of an instruction; no branch instruction is assumed in this example).
R	is a memory read cycle for an instruction.
PC	is the program counter. (The increment per bus cycle is PC=PC+1.)
stable state	is a full bus cycle.
transition state	is end of a bus cycle and start of the next one.

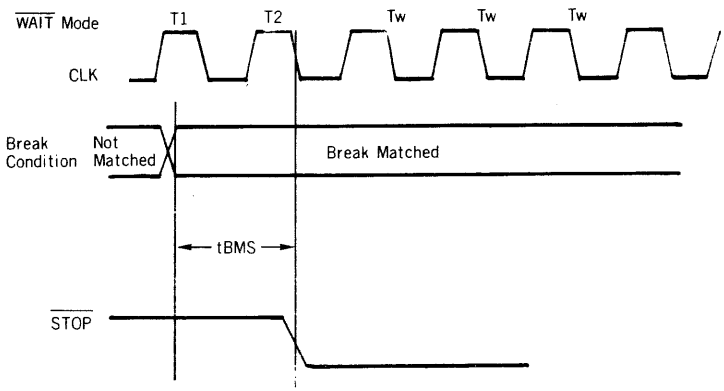
C.6 Breakpoint Timing (H5/H6) for Respective NEW MICE-II

NEW MICE-II	Break Match Timing Minimum Value (tBMS)	Stop CPU Setup Time Minumin Value		Emulation Processor	MHz
NSC800	202ns + tWS	tWS	= 50ns	NSC800	5
			100ns	NSC800-1	2
			35ns	NSC800-4	8.
Z80 $\overline{\text{WAIT}}$ Mode	249.8ns + Tsw	Tsw	= 70ns	Z80	2.5
			70ns	Z80A	4
			60ns	Z80B	6
			50ns	Z80H	8
Z80 $\overline{\text{BUSAK}}$ Mode	209.2ns + Tsb	Tsb	= 80ns	Z80	2.5
			50ns	Z80A	4
			50ns	Z80B	6
			40ns	Z80H	8
Z8	237ns			Z8612	12
ZS8	220ns			Z8884	20

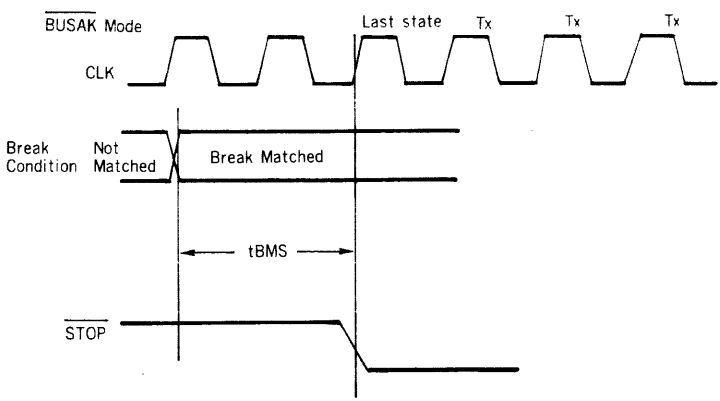
1) NSC800



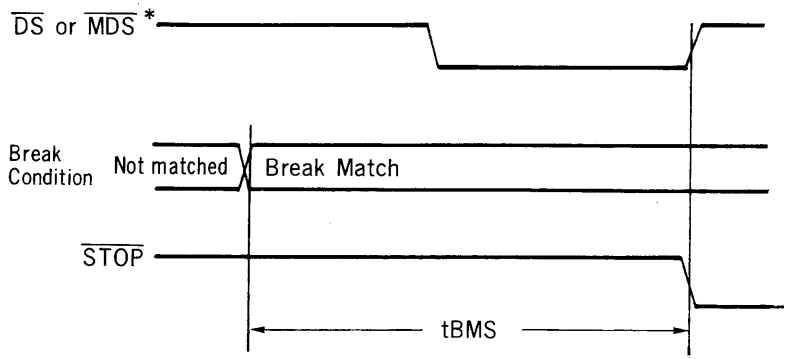
2) Z80 $\overline{\text{WAIT}}$ Mode



3) Z80 $\overline{\text{BUSAK}}$ Mode



4) Z8 & ZS8



Note: Emulation will stop at the end of the current instruction, where STOP means that the CPU will be executing "JR $" instructions.

SUEM

Section

SUEM SECTION

First Edition
March 1988

Microtek International, Inc.
No. 6, Industry East Road 3
Science-based Industrial Park
Hsinchu, Taiwan 30077, R.O.C.
Tel: (035) 772155
Tlx: 32169 MICROTEK
Fax: (035) 772598

Printed in Taiwan, ROC. March 1988

C O N T E N T S

1.0 INTRODUCTION

This document provides reference information on the operation of the SUEM.

The user of this document is assumed be reasonably familiar with the NEW MICE-II emulator family.

2.0 SPECIFICATION

2.1 DESCRIPTION

The SUEM is a high performance, soft mapped emulation memory board, that replaces an HUEM in all NEW MICE-II/S units.

2.2 MEMORY SIZE

256 Kbytes

2.3 MAPPING RSOLUTION

Three modes:
1) 128 byte resolution (address range 0 to 128K)
   (64K code, 64K data for microcontrollers)
2) 1 Kbyte resolution (address range 0 to 1M)
3) 8 Kbyte resolution (address range 0 to 16M)

2.4 NON-VOLATILE STORAGE

The memory map settings can be stored in EEPROM. This setup is automatically recalled on power up.

2.5 INDICATIONS

The SUEM provides two LED indicators

2.5.1 Run Light

This shows whether the emulation processor is in a run state.

2.5.2 Power Light

This shows that 5 Volt supply is applied.

2.6 ACCESS VIOLATION

Outputs provided for address violation are:

* Internal (emulation memory) write access
* External (target memory) write access
* Guard memory (non-existent) access

2.7 VIOLATION ACCESS SETTINGS

The Violation Access Signals generated from SUEM are located at J3 (3-pin connector).

J3 Pin Definition

Pin 1 (J3-1)	Guard access violation
Pin 2 (J3-2)	Write protect violation
Pin 3 (J3-3)	Either guard access or write protect violation

The signals are active at high level.

To enable the violation to break the program execution, J3-1 and J3-2 should be wired (using jumper wires or others) to CEP spare bits X0 and X2 respectively. Then use BPP command to set the break conditions H5, H6.

Application notice for violation access break see appendix B.

3.0 SWITCH SETTINGS

3.1 DESCRIPTION:

SW1, an 8-way DIP switch, is located at the front of the SUEM and is accessible when the cover of the NEW MICE-II is removed. This switch configures the host interface (between host and MICE) and the inter board (between CEP and SUEM).

host baud rate	S1	S2	S3
150	on	off	off
300	off	on	off
600	on	on	off
1200	off	off	on
2400	on	off	on
4800	off	on	on
9600	on	on	on
19200	off	off	off

host mode	S4	S5	S6
8 data bits	off		
7 data bits	on		
parity enable		off	
parity disable		on	
even parity			off
odd parity			on

	SUEM DIP Switch		CEP DIP Switch					
inter comms*	S7	S8	S1	S2	S3	S4	S5	S6
9600	n/u**	on	on	on	on	off	on	n/u**
19200	n/u	off	off	off	off	off	on	n/u

* Note that inter communication mode is 8 data bits, no parity. On CEP, the DIP switch should be set to 19200 (or 9600) bauds, no parity and 8 data bits.

** not used

3.2 INSTALLATION

NOTE

This section is applicable only when replacing an HUEM with SUEM in MICE-II

a) Remove the existing HUEM board from NEW MICE-II and replace with the SUEM board.

b) Reconnect the two 50-pin connectors to SUEM.

c) Remove the Interface (DTE/DCE converter) header from the CEP board. Plug-in the CSCB (CEP SUEM Communication Board) cable header into the just vacated header socket.

4.0 COMMAND REFERENCE

4.1 HELP COMMAND

***?**

---

*? is the command for SUEM help.

This command provides the user with a command summary together with a syntax guide.

EXAMPLE:

```
>*?
              SUEM   V#.#
              -----------

DIAGNOSTIC TESTS        *D
HIGH SPEED DOWNLOAD     *L[OAD]
MEMORY MAP CONTROL      *M[AP]    [ADDR1 ADDR2 I|IR|E|ER|G]|[E|D]
MAPPING RESOLUTION      *R[ANGE]  [0|1|2]
RECALL MAP              *RE[CALL]
SAVE MAP                *S[AVE]
HELP                    *?
>
```

4.2 DIAGNOSTIC TESTS COMMAND

***D**

*D is the command for SUEM diagnosis. Issuing this command will test emulation memory as well as SUEM hardware ciruits. Results are shown when diagnostic testing is completed.

After diagnostic testing, the origienal map settings, map resolution and contents in emulation memory are destroyed.

EXAMPLE:

```
>*D
Diagnostics: testing U12 & U26

Diagnostics: testing U13 & U27

Diagnostics: testing U14 & U28

Diagnostics: testing U15 & U29

Diagnostics: 256K ram installed
>
```

4.3 HIGH SPEED DOWNLOAD COMMAND

***LOAD**

***LOAD** is the command for high speed download.

After entering this command, the SUEM will accept an hex file sent via the serial port. An end of file record or an escape will terminate this mode and return the MICE prompt.

EXAMPLE:

>*LOAD <CR>

NOTE

The *LOAD command does not use the ACK/NAK protocol, but it does support hardware of software handshaking for flow control. The MICE CEP must have the ETX handshaking code installed in the firmware for this command to function.

4.4 MEMORY MAP CONTROL

***M[AP]**

***M[AP] start-address end-address memory-attrib**

***M[AP] global-controls**

This command has several options, allowing the memory map to be set, displayed or modified.

***M[AP]** is the command for memory map control. If no other parameters specified, inputting "MAP<CR>" displays the current memory map setting.

start-address is a a hexadecimal address indicating the start of a boundary as defined by the current mapping range setting. (also refer to the *RANGE command).

end-address is a hexadecimal address indicating the end of a boundary as defined by the current mapping range setting. (also refer to the *RANGE command).

memory-attrib can be one of the following:

I - internal (emulation) memory with read/write access

IR - internal (emulation) memory with read only access

E - external (target) memory with read/write access

ER - external (target) memory with read only access

G - guarded access

global-controls can be one of the following:

E - enable map settings

The current memory map settings will be used by the emulator..

D - disable map settings

The memory map settings are ignored and the emulator will use target system memory. The previously set memory map can be reenabled by typing *MAP E.

EXAMPLE:

```
>*MAP
Range: 0 to 16Mb with 8k resolution

000000 to FFFFFF     guarded              ( G)

>*MAP 0 1FFF I
>*MAP
Range: 0 to 16Mb with 8k resolution

000000 to 001FFF     internal read/write ( I)
002000 to FFFFFF     guarded              ( G)
```

```
>*MAP 100000 165FFF ER
>*MAP
Range: 0 to 16Mb with 8k resolution

000000 to 001FFF     internal read/write ( I)
002000 to 0FFFFF     guarded             ( G)
100000 to 165FFF     external read only  (ER)
166000 to FFFFFF     guarded             ( G)

>*MAP D
Map: emulation memory globally disabled
>*MAP 0 1FFF E
Map: emulation memory globally disabled
>*MAP E
Map: emulation memory is enabled
>

>*MAP
Range: 0 to 16Mb with 8k resolution

000000 to 001FFF     internal read/write  ( I)
002000 to 0FFFFF     guarded              ( G)
100000 to 165FFF     external read only   (ER)
166000 to FFFFFF     guarded              ( G)

>
```

NOTE

The SUEM halts the emulation processor when modifying the memory map, but it does not halt the emulation processor when displaying the memory map. It is up to the user to take the appropriate action when using this command.

4.5 MAPPING RESOLUTION CONTROL

***R[ANGE]**

***R[ANGE] type**

This command allows the user to display or select three different types of mapping resolution.

***R[ANGE]** is the command for memory mapping resolution control. If no other parameters specified, inputting "*RANGE<CR>" displays the current mapping resolution.

type may be one of the following:

0 - 128 byte resolution, in the range 0 to 128K (17-bit addressing). For processors with separate address space for code and data (8051, 80515, Z8, ZS8), the SUEM will use the 17th address line of the memory map to signify data memory. This gives us 0 to 64K for code and 0 to 64K for data (mapped from 64K to 128K). Processors having 16 address lines will only be able to access the first 64K of emulation memory.

1 - 1K resolution in the range 0 to 1M bytes (20-bit addressing).

2 - 8K resolution in the range 0 to 16M bytes (24-bit addressing).

The emulation processor will be halted after this command is executed.

EXAMPLE:

```
>*RANGE
Rang: 0 to 16Mb with 8k resolution
>*RANGE 1
Range: 0 to 1Mb with 1k resolution
>*RANGE 0
Range: 0 to 128K with 128 byte resolution
```

NOTE

When the mapping resolution is changed, all the memory is mapped GUARDED.

If the emulation processor's memory range is 0 to 16M byte (24-bit addressing) and the user want to get a smaller mapping resolution, user can set a lower type of mapping resolution (0 or 1) with *RANGE command, but the memory will overlap.

EXAMPLE:

```
>*R 1
Range: 0 to 1Mb with 1k resolution
>*M 0 FFF I
>M FF0 FFF AAAA
>M FF0 FFF
       0000 0002 0004 0006 0008 000A 000C 000E     ASCII-CODE
000FF0 AAAA AAAA AAAA AAAA AAAA AAAA AAAA AAAA  ................
>M 100FF0 100FFF
       0000 0002 0004 0006 0008 000A 000C 000E     ASCII-CODE
100FF0 AAAA AAAA AAAA AAAA AAAA AAAA AAAA AAAA  ................
>
```

4.6 RECALL MEMORY MAP

***RECALL**

This command allows the user to recall a previously saved memory map from the EEPROM. The current memory map will be destroyed.

EXAMPLE:

```
>*RECALL
>
```

NOTE

The emulation processor will be halted after executing this command.

4.7 SAVE MEMORY MAP

***S[AVE]**

SAVE is the command for saving memory map settings to EEPROM.

EXAMPLE:

```
>*SAVE
.............
>
```

NOTE

The *SAVE command will take 20 seconds (worst case) to execute.

5.0 APPLICATION NOTES

5.1 BINARY DOWNLOAD

Applying an 8MHz IBM-PC/AT as the target, download speed under USD (software V2.3 and later versions) at 64K bytes record length is illustrated below.

Communication Protocol	19200 baud None parity 8 data bits 1 stop bit	19200 baud Even parity 7 data bits 1 stop bit
Transmission Format	Binary	Hex
Download Speed	51 seconds	2 min. 38 sec.

Note that while using binary transimission format, communication protocol should be set to none parity and 8 data bits.

5.2 LIMITATION OF SUEM FIRMWARE V1.0 AND V1.1

To get proper communication between SUEM and host, baud rate should not less than 2400.

APPENDIX A

LIMITED WARRANTY; Service

Microtek International Inc. (Microtek) warrants **Super Universal Emulation Memory or SUEM** (the Product) and the user's manual for the Product to be free from physical defects for a period of twelve (12) months from the date of the original retail purchase. This warranty applies only to the original retail purchaser who bought these Products from an authorized Microtek representative. This warranty is void if the Product is damaged by improper or abnormal use or by accident, if the Product is altered or modified in any way, or if any attempt is made to repair the Product without authorization from Microtek.

If you find a Product to be defective, contact your authorized Microtek representative or Microtek. When you receive authorization to do so, return the Product as directed; include proof of purchase and purchase date. Microtek will correct physical defects in Products under warranty at no charge to you by repairing or, at its option, replacing such Products. THIS IS THE SOLE AND EXCLUSIVE REMEDY AVAILABLE FOR BREACH OF WARRANTY OR UNDER ANY OTHER LEGAL THEORY WITH RESPECT TO MICROTEK PRODUCTS. Product repairs not covered by warranty, and Product updates, are provided at a set rate.

ALL OTHER WARRANTIES AND REPRESENTATIONS, ORAL OR WRITTEN, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, ARE EXCLUDED AND DO NOT AP-

PLY. Except as set forth in this limited warranty, the purchaser assumes the risk as to these Products' quality, accuracy, design, and performance.

It is solely the purchaser's responsibility to determine the suitability of these Products for each particular application. Microtek Products are in all events not suitable, and are not authorized, for use in connection with life support devices or systems, or in other devices or systems potentially injurious to life or health.

IN NO EVENT WILL MICROTEK BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY BREACH OF WARRANTY OR UNDER ANY OTHER LEGAL THEORY, even if advised of the possibility of such damages. Microtek is thus not liable for lost profits or goodwill; downtime; damage or destruction of any program, data, equipment, or other property; costs of recovering, reprogramming, or reproducing any program, data, or equipment; personal injury or loss; or any other damages.

Except as required by law, no representative, agent, or employee of Microtek is authorized to commit Microtek to warranties, representations, or obligations inconsistent with or in addition to those set forth in this limited warranty.

Please complete and return the registration section of the warranty card (provided in the packing box) to Microtek.

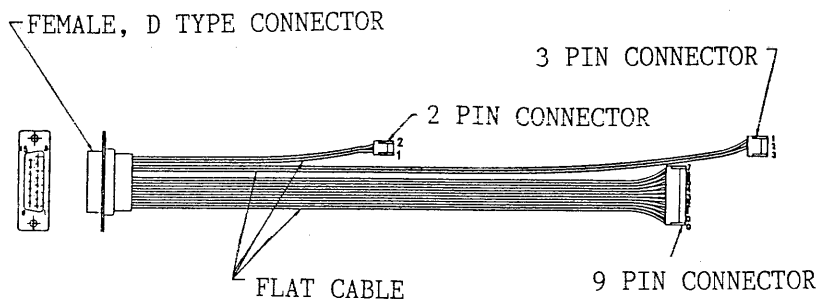
APPENDIX B

PROVIDING THE VIOLATION ACCESS BREAK

---

1. In order to provide a violation access break for New MICE-II/S, the trace cables have to be modified as follows:

   1) Connect the violation access signals (SUEM J3) to New MICE-II case trace bits port (see figure 1).



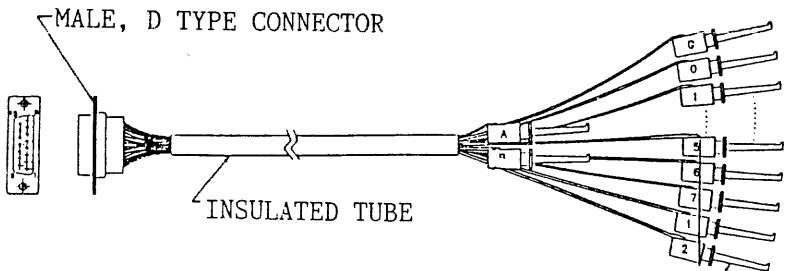
D TYPE CONN.	.098" CONN.	
FEMARE,15PIN	8PIN&2PIN&3PIN	WIRE COLOR
PIN NO.	PIN NO.	
1	G (GND)	BLACK
9	0 (8 PIN)	BROWN
2	1 (8 PIN)	RED
10	2 (8 PIN)	ORANGE
3	3 (8 PIN)	YELLOW
11	4 (8 PIN)	GREEN
4	5 (8 PIN)	BLUE
12	6 (8 PIN)	PURPLE
5	7 (8 PIN)	GRAY
15	1 (2 PIN)	BLACK
8	2 (2 PIN)	BROWN
13	1 (3 PIN)	BLACK
6	2 (3 PIN)	BROWN

Figure 1

* 3-pin connector pin 1 (connecting to SUEM J3-1) connects to D type (female) connector pin 13.
* 3-pin connector pin 2 (connecting to SUEM J3-2) connects to D type (female) connector pin 6.

2) Connect two IC clips (marked A and B with shorter wire) to D type (male) connector pin 13 and 6 on external trace cable (see figure 2).

* IC clip A connect to D type (male) connector pin 13.
* IC clip B connect to D type (male) connector pin 6.



D TYPE CONN. , 15 PIN		IC CLIP , 40MM LG	
PIN NO.	WIRE COLOR	PIN NO.	COLOR
1	BLACK	G (GND)	BLACK
9	BROWN	0 (8 PIN)	RED
2	RED	1 (8 PIN)	RED
10	ORANGE	2 (8 PIN)	RED
3	YELLOW	3 (8 PIN)	RED
11	GREEN	4 (8 PIN)	RED
4	BLUE	5 (8 PIN)	RED
12	PURPLE	6 (8 PIN)	RED
5	GRAY	7 (8 PIN)	RED
15	BLACK	1 (2 PIN)	BLACK
8	BROWN	2 (2 PIN)	BLACK
13	BLACK	A (2 PIN)	RED
6	BROWN	B (2 PIN)	RED

Figure 2

2. If user wants to use the violation access break feature, follow the procedure listed below:

   1) Connect the red IC clip 0 (trace bit X0) to IC clip A.
   2) Connect the red IC clip 2 (trace bit X2) to IC clip B.
   3) Set the BPP external hardware breakpoint H5 and H6 as follows:

```
> H5 X1          ;for guard access violation
> H6 X1          ;for write protect violation
```

   If an attempt is made to write into Read-only area (defined as "IR" or "ER" in the MAP command) while emulation processor running, this message will display -

   "PROGRAM BROKE AT BREAKPOINT-6"

   Likewise, if a non-existent area (defined as "G" in MAP command) is accessed, this message will display -

   "PROGRAM BROKE AT BREAKPOINT-5"

Notes: 1. Only the delivery of New MICE-II/S or/H includes these two trace cables (refer to fig. 1 and 2), SUEM single package does not.

2. Only New MICE-II/S provides the violation access break.

3. The PCB revision (Rev-B) of SUEM will move the position of J3 connector near the U63. This modification will shorten the wire length between the J3 and D type (female) connector.

MICROTEK