

Applied Microsystems Corporation



## 6800/6802 Series Microprocessors



5020 148th Ave. N.E.  
Redmond, WA 98052  
or  
P.O. Box C-1002  
Redmond, WA 98073-1002

Copyright © 1984. All rights reserved

**EM-186 Diagnostic Emulator\***

920-10649  
1-84

\*Diagnostic Emulator is a Trademark of Applied Microsystems Corporation

---

*Applied Microsystems Corporation has made every effort to document this product accurately and completely. However, Applied Microsystems assumes no liability for errors or for any damages that result from use of this manual or the equipment it accompanies. Applied Microsystems reserves the right to make changes to this manual without notice at any time.*

*Because this configuration of the EM-186 Diagnostic Emulator is intended for use in developing, debugging, and testing Motorola 6800/6802 Series microprocessor-based systems, it is presumed that the user is familiar with the terminology of the 6800/6802 Series microprocessor.*

---



## Table of Contents

### Section 1 - Introduction

1-1	System Concept	1-2
1-2	Transparency	1-2
1-3	Warranty	1-2
1-4	General Specifications	1-3

### Section 2 - EM-186 Components

2-1	Operator's Station	2-2
2-2	Emulator Probe	2-2
2-3	Keyboard	2-2
2-4	Diagnostic EPROM Socket	2-2
2-5	Display Panel	2-2
2-6	Trace Memory	2-2
2-7	Back Panel Controls and Connectors	2-5
2-8	RAM Overlay	2-5
2-9	Disassembler	2-6

### Section 3 - Basic Operating Instructions

3-1	Operating Voltage	3-2
3-2	Safety Information	3-2
3-3	Connection to Target Equipment	3-2

### Section 4 - EM-186 Functions

4-1	Execution Control	4-2
4-1.0	RESET Key	4-2
4-1.1	RUN Key	4-2
4-1.2	RUN BKPT Keyswitch	4-3
4-1.3	STEP Keyswitch	4-3
4-1.4	Breakpoint System	4-5
4-1.5	Trace Memory	4-8
4-2	Examination and Alteration of MPU Registers	4-10
4-3	Examination and Alteration of Memory Locations	4-11

### Section 5 - RAM Overlay

5-1	Overview	5-2
5-2	Installation	5-3
5-3	Controls	5-4
5-4	Uploading/Downloading	5-6

### Section 6 - Disassembly

6-1	Overview	6-2
6-1.1	Operation Preparation Procedures	6-2
6-2	Format Definition	6-7



**Section 7 - Built-In Diagnostic Functions**

7-1	Group A: Memory Tests	7-3
7-2	Group B: Oscilloscope Loops	7-6
7-3	Group C: Memory Load and Dump	7-8
7-4	Group D: Miscellaneous	7-13
7-5	Group E: Change Default Parameters	7-16
7-6	Group F: Internal Operations	7-17

**Section 8 - User Implemented Code Functions**

8-1	Overview	8-2
8-2	Internal Environment	8-3
8-2.1	ROM	8-4
8-2.2	Front Panel EPROM Socket	8-4
8-2.3	Scratchpad RAM	8-4
8-2.4	I/O Devices	8-4
8-3	Entry to User Code Functions	8-15
8-4	Introspection Mode	8-16
8-4.1	Code F	8-17
8-5	Getting To and From the Target System	8-17
8-5.1	Examine and Store	8-17
8-5.2	Pause to Run	8-18
8-5.3	Run to Pause	8-18
8-5.4	Re-Entry Jump	8-19
8-6	User Accessible Subroutines	8-20
8-7	Interrupts	8-25
8-8	Code Function Examples	8-25

**Section 9-Supplementary Information**

9-1	Auxiliary Connector	9-2
9-2	Option Switches	9-4
9-3	Serial Interface	9-5
9-4	Upload/Download Protocol	9-6
9-5	External Breakpoint	9-8
9-6	Trace Hold	9-9
9-6.1	Window Mode	9-9
9-6.2	Hold Trace on Breakpoint	9-9
9-6.3	Selective Trace	9-9
9-7	Signature Analysis	9-10
9-8	Soft Shutdown	9-13

**Section 10 - Maintenance and Troubleshooting**

10-1	Maintenance	10-2
10-1.1	Power Supply	10-2
10-1.2	Cables	10-2
10-1.3	Probe Tip Assembly	10-2
10-2	Troubleshooting	10-3

**Index**

# SECTION 1

---

---

---

## INTRODUCTION

---

- 1-1 System Concept**
  - 1-2 Transparency**
  - 1-3 Warranty**
  - 1-4 General Specifications**
-



## 1-1 SYSTEM CONCEPT

The EM-186 Diagnostic Emulator is a microprocessor test and diagnosis instrument designed to emulate the 6800, 68A00, or 68B00 microprocessors. An alternate pod and cable assembly enables the unit to emulate the 6802, 68A02, 68B02 or 6808 MPU. The Diagnostic Emulator consists of an Operator's Station with Keyboard and Display Panel and an Emulator Pod and Cables for connection to the user's system. The EM-186 is fast and easy to use and includes many diagnostic capabilities for troubleshooting problems in the user's system.

Figure 1-1.1



## 1-2 TRANSPARENCY

The EM-186 Diagnostic Emulator is transparent to the normal operation of the target system in that emulation is in real-time, with no additional processor cycles required as a result of the emulation process. There are no target system addresses needed or used by the EM-186 and there are no programs or other software objects that are required to be in the target address space. As a consequence of this transparency, the user should not experience difficulties in using the EM-186 Diagnostic Emulator with his system even if there are critical software timing constraints in his system.

## 1-3 WARRANTY

Applied Microsystems Corporation (AMC) warrants that the articles furnished hereunder are free from defects in material and workmanship and perform to applicable published AMC specifications for one year from date of shipment. This warranty is in lieu of any other warranty expressed or implied. In no event will AMC be liable for special or consequential damages as a result of any alleged breach of this warranty provision. The liability of AMC hereunder shall be limited to replacing or repairing, at its option, any defective units which are returned F.O.B. AMC's plant. Equipment or parts which have been subject to abuse, misuse, accident, alteration, neglect, unauthorized repair or installation are not covered by warranty. AMC shall have the right of final determination as to the existence and cause of defect. When items are repaired or replaced, the warranty shall continue in effect for the remainder of the warranty period, or for 90 days following date of shipment by AMC, whichever period is longer.

### 1-4 GENERAL SPECIFICATIONS

#### Input Power

90 to 140 Vac  
 60 Hz  
 less than 50 watts

#### Optional

180 to 280 Vac  
 50 Hz  
 less than 50 watts

#### Physical

##### Operators' Station

Width: 292mm (11.5 inches)  
 Height: 117mm (4.6 inches)  
 Depth: 356mm (14 inches)

##### Target System Connection (Ribbon Cable)

Total Length (including Pod):  
 1.5M (58 inches)

##### Emulator Cable Pod

Length: 157mm (6.2 inches)  
 Width: 90mm (3.6 inches)  
 Depth: 33mm (1.3 inches)

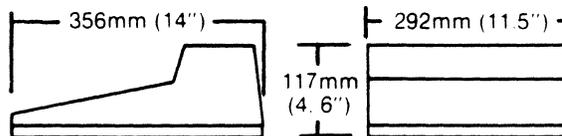
##### Total Weight:

5.4 Kg (12 lbs)  
 Shipping 7.7 Kg (17 lbs)

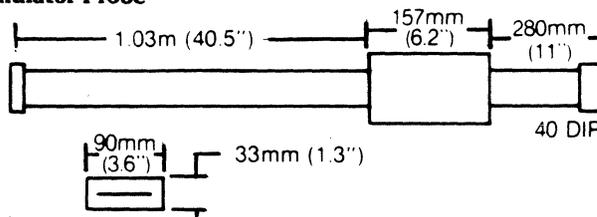
##### Environmental:

Operating Temperature: 0°C to 40°C (32°F to 104°F)  
 Storage Temperature: -40°C to 70°C (40°F to 158°F)  
 Humidity: 5% to 95% RH noncondensing

##### Operator's Station



##### Emulator Probe





# SECTION 2

---

---

---

## EM-186 COMPONENTS

---

- 2-1 Operator's Station**
  - 2-2 Emulator Probe**
  - 2-3 Keyboard**
  - 2-4 Diagnostic EPROM Socket**
  - 2-5 Display Panel**
  - 2-6 Trace Memory**
  - 2-7 Back Panel Controls and  
Connectors**
  - 2-8 RAM Overlay**
  - 2-9 Disassembler**
-



**2-1 OPERATOR'S STATION**

The EM-186 Operator's Station consists of a Keyboard, Display Panel, Diagnostic EPROM Socket, Back Panel Controls and Connectors. It contains most of the system electronics, including the emulation control circuitry, Trace Memory, Breakpoint Comparators, plus control firmware with preprogrammed test routines. A RAM Overlay option may be included. See Figure 2-1.1.

**2-2 EMULATOR PROBE**

Two Emulator Probes are available: The EP-6800 for the 6800, 68A00 and 68B00 microprocessors and the EP-6802 for the 6802, 68A02, 68B02 and 6808. The Probe contains the MPU and associated circuitry and buffers. It connects to the Operator's Station via 40-inch ribbon cables and to the target system MPU socket via 11inch ribbon cables and a 40-contact DIP connector.

**2-3 KEYBOARD**

The Keyboard has 32 keyswitches divided into four groupings: Processor Control, Mode Select, Subfunction Control and Data Entry.

**2-4 DIAGNOSTIC EPROM  
SOCKET**

A low insertion force EPROM socket to accept EPROMs compatible with Intel 2716 or 2732 types (single +5 power supply and Intel pinout). The user may create his own system test and diagnosis routines, program the EPROM with these routines, insert the EPROM into the EM-186 front panel socket and then execute the routines in a convenient manner from the EM-186 Keyboard. See Section 8: USER IMPLEMENTED CODE FUNCTIONS.

**2-5 DISPLAY PANEL**

The Display Panel consists of LED dot-matrix address and data displays and of individual LED indicators. Address and data information are displayed in hexadecimal notation. The individual indicator LEDs are divided into five groupings: Fault indicators (CLK, RESET) show loss of system clock or a continuous RESET condition; Machine Cycle indicators (FETCH, BKPT, VCTR, IRCY, IRQ, NMI, RD, WR) readout the control bus and other information acquired during target program execution; the microprocessor condition code bits (H, I, N, Z, V, C) are also displayed on these indicators; MPU Status indicators (Illegal Instruction, RE, BA, TSC, MR, Halt, Pause) show the condition of the emulated target system MPU; Breakpoint Enable (BKPT ENA) is illuminated if the Breakpoint System is enabled.

**2-6 TRACE MEMORY**

The Trace Memory is a 254-word by 32-bit memory that captures information from each bus cycle of the emulated target system microprocessor. The information recorded is: the 16 address bits, 8 data bits, MPU read and write signals, the type of bus cycle (i.e., opcode fetch, Vector, Irrelevant Cycles, Interrupt Request/Non Maskable Interrupt) and the Breakpoint Comparator.

# SECTION 2

## EM-186 COMPONENTS

Figure 2-1.1. Operator's Station.

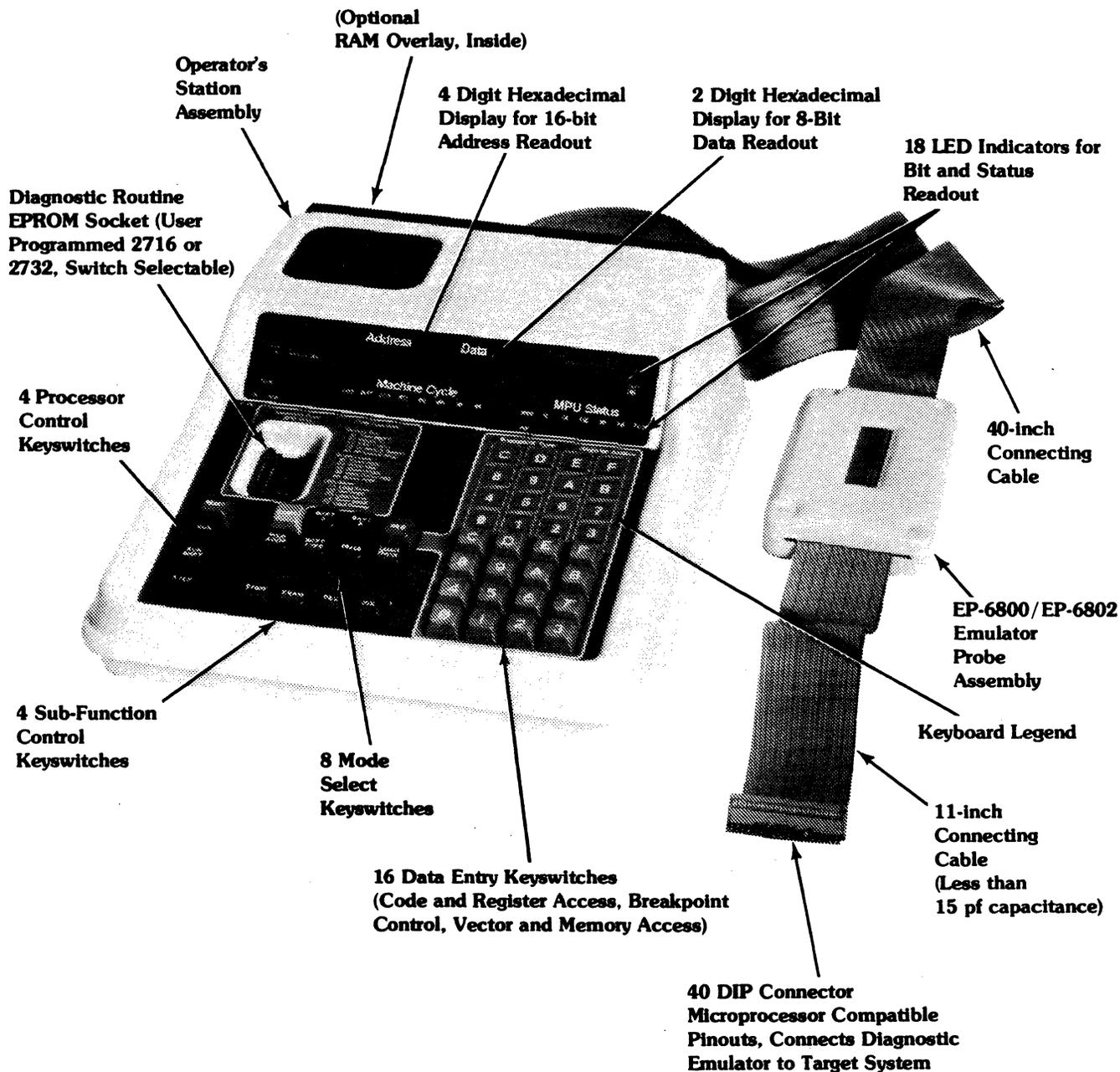
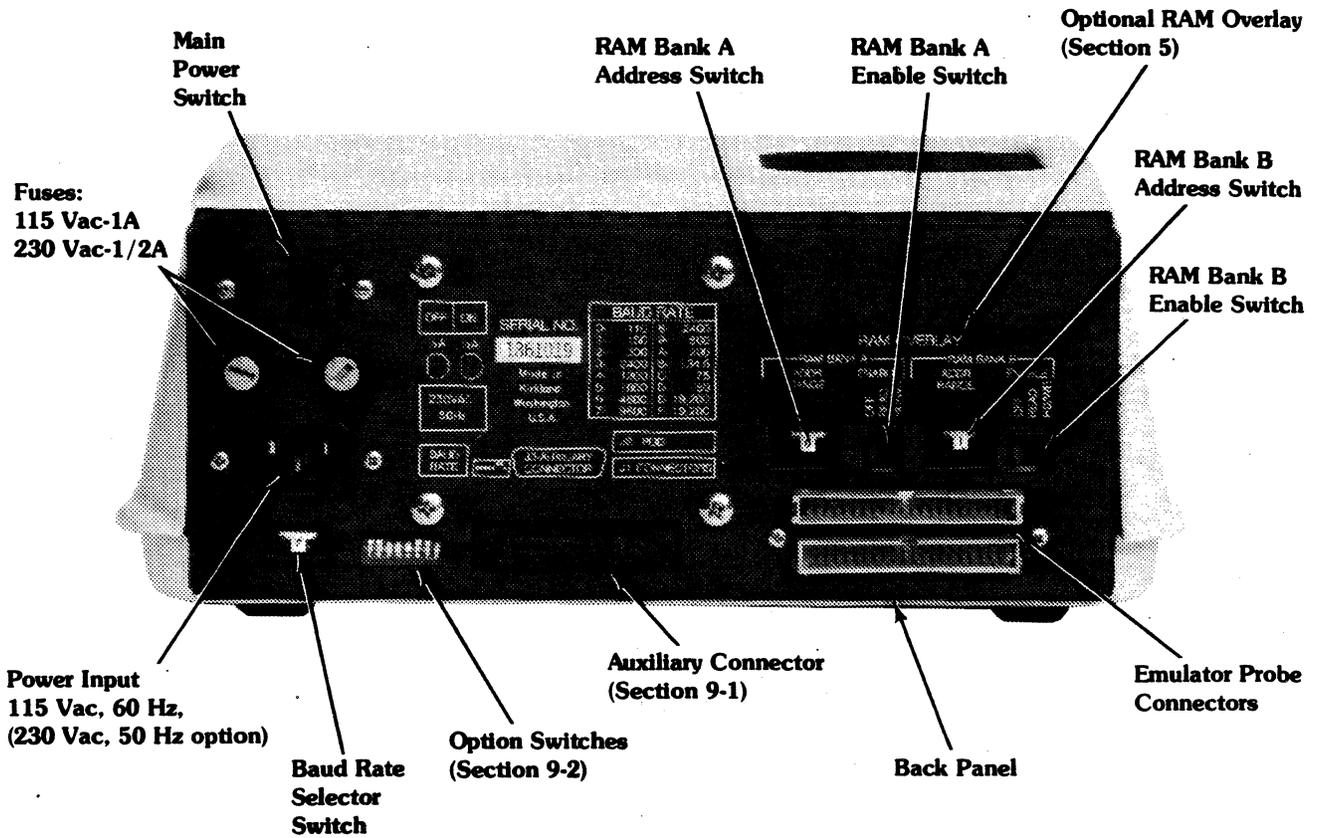


Figure 2-7.1. Back Panel.



### 2-7 BACK PANEL CONTROLS AND CONNECTORS

The Back Panel of the EM-186 includes the controls and various connectors used to connect the Diagnostic Emulator with power, the Emulator Probe and other external equipment:

#### **Main Power Switch**

Controls the primary power to the unit.

#### **Baud Rate Selector Switch**

A 16-position switch is used to control the transmission rate of serial data flow between the Diagnostic Emulator and peripheral equipment. The baud rate selection options are visible on the Back Panel template shown in Figure 2-7.1.

#### **Auxiliary Connector**

A 25-pin, D subminiature female connector. It provides RS-232C signals and additional control signals to auxiliary equipment (i.e., signature analyzer, oscilloscope, target system, development system). See Section 9-1.

#### **Option Switches**

These switches control characteristics of the EM-186 RESET circuitry and communications interface. The normal switch positions for most users are shown in Section 9-2. The EM-186 is shipped from the factory in this configuration. Alternative positions are discussed in Section 9-2.

#### **RAM Overlay Bank A and Bank B Address Switches**

Two 16-position switches are used to select the address range to which the A and B blocks of enabled overlay memory responds. See Section 5, RAM Overlay.

#### **RAM Overlay Bank A and Bank B Enable Switches**

Two 3-position toggle switches control the A or B block of overlay memory. The left position (OFF) disables a memory block, effectively removing it from the system. The center position (READ) enables the memory block for read-only operations (read-only-memory simulation). The right position (RD/WR) enables a memory block for both read and write operations.

### 2-8 RAM OVERLAY

The EM-186 may be configured to include optional overlay memory. This feature consists of 8K bytes of 200 nsec static memory that is divided into two independent 4K byte blocks. Each block may be enabled as read-write memory, used as read only memory or disabled. Back Panel switches are used to adjust each memory block to reside in any one of 16 address blocks in the target address space. When a memory block is enabled, it is mapped into the target address space, overlaying the user's system in the address block selected. The overlay memory may be loaded from the target system memory, front panel EPROM or external device by executing the appropriate Code Function. See Section 5, RAM Overlay.



## 2-9 DISASSEMBLER

The EM-186 may be configured with an optional firmware package that provides for formatting and output of system information to an ASCII terminal device with RS-232C interface such as a CRT or hard copy terminal. The disassembly firmware extracts information from the EM-186 Trace Memory and emulation processor registers, formats the data for display with instruction opcodes given in standard Motorola mnemonic form (JMP, ADD, PSH, etc.) and outputs the data through the serial port. See Section 6.

# SECTION 3

---

---

---

## BASIC OPERATING INSTRUCTIONS

---

- 3-1 Operating Voltage**
  - 3-2 Safety Information**
  - 3-3 Connection to Target Equipment**
-



### 3-1 OPERATING VOLTAGE

The EM-186 Diagnostic Emulator is normally supplied for operation from 90 to 140 Volts AC at 58 to 62 Hz line. The unit is also available for operation from 180 to 280 Volts AC at 48 to 52 Hz line if specified at time of order. The EM-186 uses a regulating transformer that also has the advantage of providing good blocking of conducted noise that may be present on the power input to the unit.

### 3-2 SAFETY INFORMATION

The EM-186 is supplied with a 3-wire cord with a 3-terminal polarized plug for connection to the power source and protective ground. The ground terminal of the plug is connected to the metal chassis parts of the instrument. Electric-shock protection is provided if the plug is connected to a mating outlet with a protective ground contact that is properly grounded.

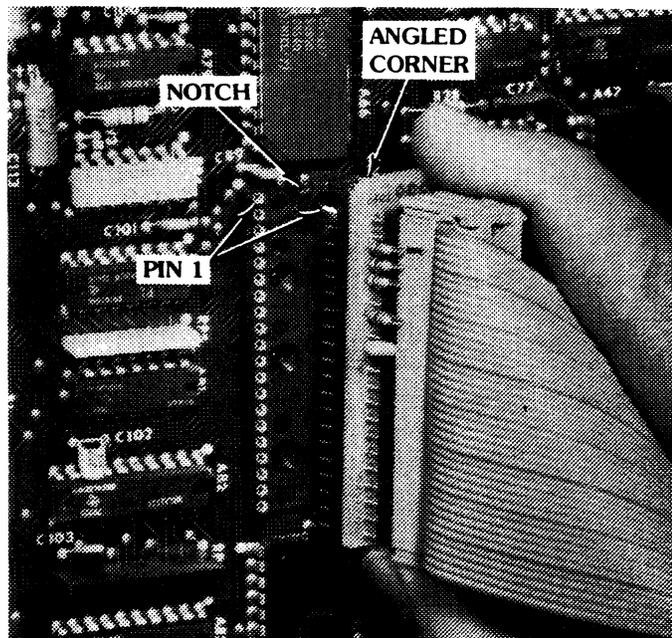
The internal (logic) ground of the EM-186 is not connected to the protective ground, but floats to the same potential as the equipment to which the unit is connected. The user is cautioned that it is conceivable that grounding conflicts could occur if the EM-186 is connected to two different items of equipment with differing ground potentials such as the target equipment and a RS-232C terminal.

### 3-3 CONNECTION TO TARGET EQUIPMENT

Connect the EM-186 Diagnostic Emulator to the target system by removing the microprocessor from its socket in the user's system and plugging in the 40-pin plug of the EM-186. (See Figure 3-3.1.)

#### CAUTION: NOTE CORRECT PIN 1 ORIENTATION

Figure 3-3.1. Installing 40-pin plug.



# SECTION 3

## BASIC OPERATING INSTRUCTIONS

---

---

Apply power to the EM-186 and the target system after the unit is properly connected to the target circuitry. Once power is applied to the Diagnostic Emulator and the clock begins operating, it performs a Power-on-reset operation during which the following functions are performed:

1. Reset MPU.
2. Clear Trace Memory and MPU Registers.
3. Clear the program to the user starting address, and display.
4. The Diagnostic Emulator awaits further input from the operator.

After the EM-186 is connected to the target system and power is applied to both the EM-186 and the target system, perform the following checks to verify that the unit is operating correctly:

1. The Clock Fault and Reset Fault indicators are not illuminated. This means that the system clock oscillator is operating and is being received by the EM-186 and that there is not a continuous RESET signal from the target system.
2. The PAUSE indicator should be illuminated. This indicates that no target program is executing and that the EM-186 is awaiting operator commands.
3. At power on time, the ADDRESS Display should be at the user vector. If it does, the EM-186 internal control program is operating.

If all is well, proceed to operate the Diagnostic Emulator as appropriate. See Section 4 for details of EM-186 functions.



# SECTION 4

---

---

---

## EM-186 FUNCTIONS

---

- 4-1 Execution Control**
  - 4-2 Examination and Alteration  
of MPU Registers**
  - 4-3 Examination and Alteration  
of Memory Locations**
-



A basic function of the EM-186 is to emulate the target system microprocessor. Effectively, the Diagnostic Emulator is a pin-compatible functional replacement for the microprocessor in the target system. The unit is designed to meet the timing specifications of the emulated processor and to minimize the increase in electrical loading of the user's system.

The EM-186 is always in one of two modes: RUN or PAUSE. If in the RUN mode, the EM-186 is emulating the target system microprocessor and executing the target system program at full system speed. The Trace Memory will be active (unless inhibited by external control) and all bus cycles of the emulated microprocessor are recorded for possible later display. In the PAUSE mode, emulation of the target system microprocessor is suspended and the operator is able to perform other functions such as manually examining or altering memory locations or internal registers of the emulated microprocessor; the operator may also review the history of the target program execution from the Trace Memory or execute one of the Code Function routines.

#### 4-1 EXECUTION CONTROL

The operator controls the EM-186 primarily through the Operator's Station Keyboard. Keyswitch groupings are designed for easy understanding and convenient use. The EM-186 display provides the operator with information about program execution, MPU status and EM-186 conditions. Table 4-1 defines the Display Panel indicators.

##### 4-1.0 RESET Keyswitch

The red RESET Keyswitch always resets the microprocessor and initializes the EM-186 in the PAUSE mode. At this time the Address Display shows the user vector starting address. The program starting address may be changed at this time by entering digits with the hexadecimal keyswitches, or the current program starting address may be used.

The option switches accessible at the Back Panel of the EM-186 may be used to set up one of several options concerning the RESET circuitry of the EM-186 and the target system. See Section 9-2.

##### 4-1.1 RUN Keyswitch

Pressing the RUN Keyswitch causes the EM-186 to execute the target program beginning at the user vector address or continuing from the last instruction executed. Execution is at full system speed with no extra wait states beyond those commanded by the target system. The activity of the executing program is recorded continuously in the Trace Memory. It is also possible to obtain a general view of the program activity by watching the displays. For example, it is possible to tell if the program is in a tight loop or ranging widely in the program address space by observing changes in the Address Display.

### 4-1.2 RUN BKPT Keyswitch

This Keyswitch starts the EM-186 running the target program in real time, as does RUN, except the breakpoint-stop circuitry is enabled. If a breakpoint is detected, the EM-186 will pause (after finishing the instruction cycle), and the display will show the cycle where the breakpoint was detected. Pressing RUN BKPT again will cause execution to resume until the breakpoint is again detected. The breakpoint-stop circuitry may be disabled during program execution by pressing RUN.

### 4-1.3 STEP Keyswitch

Pressing the STEP Keyswitch while the program is running causes the program execution to stop. The displays at this point show the operation code fetch cycle of the last instruction executed, with the address, op-code (data) and control signals visible. When the Diagnostic Emulator stops executing the target program the following events take place:

1. The processor stops executing the target program.
2. The processor registers are saved in internal scratch pad memory and are accessible for display or alteration.

The operator, in effect, freezes the target program execution at the point reached when STEP was depressed. The operator then has several options:

1. Continue executing the program at full speed by pressing RUN.
2. Continue executing the program one instruction at a time by pressing STEP for each additional instruction execution.
3. Examine or change the contents of any of the processor registers.
4. Examine any memory location, and if the location is writable, store new data in it.
5. Review the last 254 bus cycles performed by the processor by decrementing through the Trace Memory.

The state of the target program is not changed by any of these operations (except as purposely altered by the operator) and program execution may be continued from the point where it stopped.

The program may be executed one instruction at a time by pressing STEP once for each instruction. If STEP is pressed and held down, the Diagnostic Emulator begins stepping at about seven instructions per second. The step rate then accelerates gradually from 7 steps per second to about 75 steps per second. Execution stops again if the keyswitch is released.



Table 4-1. Display Panel Indicators

**FAULT GROUP**

ILLUMINATES IF:

- CLK Target system clock not operating.  
Target system clock is low in frequency.  
EM-186 not connected to target system.
- RESET Processor and Diagnostic Emulator held in Reset by a low on the RESET terminal of the microprocessor socket.

**MACHINE CYCLE GROUP**

ILLUMINATES IF:

- FETCH Displayed machine cycle is the op-code fetch cycle of an instruction.
- BKPT Breakpoint conditions set-up for an output from the breakpoint circuitry were satisfied during the displayed machine cycle.
- VCTR Vector indicates a reading of any one of the eight vector addresses.
- IRCY Irrelevant Cycle. Memory accesses which are not required in the execution of the instruction.
- IRQ Interrupt Request. Indicates the presence of a low state at the IRQ input of the MPU.
- NMI Non-Maskable Interrupt. Indicates the presence of a low state at the NMI input of the MPU.
- READ Indicates a valid memory read cycle.
- WRITE Indicates a valid memory write cycle.

**CONDITION CODES**

ILLUMINATES IF:

- H Half-Carry bit is true.
- I Interrupt mask bit is true.
- N Negative bit is true.
- Z Zero bit is true.
- V Overflow bit is true.
- C Carry bit is true.

### MPU STATUS

ILLUMINATES IF:

Illegal Instr	Indicates that the MPU has attempted to execute or has fetched an illegal op-code such as 3C, 3D, 9D or DD.
RE	RAM Enable. Illuminates if the 6802 RAM is enabled.
BA	Bus Available. Indicates that the MPU is ready for DMA or memory refresh.
TSC	Tri-State Control. Indicates that the MPU TSC pin is in a logic 1 state.
MR	Memory Ready. The MPU clock is stopped for slow memory.
Halt	The presence of a logic 0 on the HALT pin of the MPU.
Pause	Real-time emulation of the target program is suspended and the Diagnostic Emulator is awaiting another command.
BKPT ENA	Breakpoint Enable. Illuminates if the RUN BKPT detect circuitry is armed.

#### 4-1.4 Breakpoint System

The Diagnostic Emulator incorporates a Regional/Relational breakpoint generation system to enable the user to monitor the operation of his program and to stop execution of his program when desired. The EM-186 contains two independent address comparators. Each of these comparators continuously monitors the 16-bit address bus of the microprocessor. In addition, each comparator may be qualified to respond to read cycles only, to write cycles only, or to both read and write cycles.

It is also possible to configure the breakpoint system so that a specified relationship must hold between the A and B breakpoint comparators before PAUSE occurs. The relationships that may be specified are the following:

1.      A or B            Break if condition A or condition B (or both) is found.
2.      A then B.        Break if condition A is found followed some time later by condition B.
3.      A ↔ B            Break if any address in the range from A to B (inclusive) is found.
4.      ←A—B→            Break if any address outside of the range from A to B is found. (Including addresses A and B.)



Table 4-2. Breakpoint Qualifiers

0 - Disable	C - A or B (Default)
1 - Memory Read	D - A then B
2 - Memory Write	E - Range A to B
3 - Memory Read/Write	F - Range outside A to B

The various breakpoint possibilities are set up by simple keystroke sequences. The breakpoint address and the breakpoint qualifiers may be changed independently of each other any time the emulator is in the pause mode.

Some examples of these sequences follow.

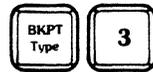
**EXAMPLE 1:**

Set up breakpoint comparator A to respond to read or write cycles at address 4300<sub>16</sub>; disable comparator B.

**KEYSTROKE SEQUENCE:**



Set breakpoint address.



Set qualifier 3 (memory R/W).



Enable BKPT B.



Set qualifier 0 (disable).

On power up, the EM-186 sets the qualifiers for both breakpoint comparators for the A OR B relation (comparators operating independently of each other) and the memory read/write qualifier. The address to which each comparator is initialized is 0000<sub>16</sub>. In the preceding example it was not necessary to alter the relationship holding between the two comparators, so the default A OR B relationship was not altered.

**EXAMPLE 2:**

Set up breakpoint comparator A to respond to read cycles only at memory address  $8A72_{16}$ , and breakpoint comparator B to respond to write cycles only at MEM. ADDR.  $13_{16}$ .

KEYSTROKE SEQUENCE:

     Set A breakpoint address.

  Set A qualifier to 1 (Memory Read).

     Set B breakpoint address.

  Set B qualifier to 2 (Memory Write).

When the breakpoint circuitry is set up as desired, program execution may be started using RUN BKPT. The function RUN BKPT is the same as the function of RUN except that when the breakpoint condition occurs, program execution stops after completing the instruction cycle. It is also permissible to start program execution using the RUN Key, and then later arm the breakpoint-stop circuitry by depressing RUN BKPT even while the target program is executing. Breakpoints may be disabled while the target program is executing by depressing RUN. The BKPT ENA indicator on the display shows the current breakpoint enable status of the emulator.

**EXAMPLE 3:** Set up write-only breakpoint range from  $2000_{16}$  to  $4307_{16}$ .

KEYSTROKE SEQUENCE:

     Set A to range beginning ( $2000_{16}$ ).

   Set qualifier to 3 (Memory R/W) and E (Range A to B).

     Set B to range end ( $4307_{16}$ ).



In this example, two qualifiers were entered: one to specify that write cycle detection was desired and a second to specify that the A and B comparators are to work together to define a continuous range. With the specifications made as shown, the breakpoint circuitry will respond to any write cycle to any address in the range of  $2000_{16}$  to  $4307_{16}$ . Note that it was not necessary to specify any qualifiers for the B comparator; this is because the two comparators are linked together to provide address range detection and the qualifiers entered for A apply also to B.

Table 4-3 shows the permissible breakpoint combinations.

	C A or B	D A then B	E A↔B	F ←A-B→
0 DISABLE	✓	✓	✓	✓
1 MEM RD	✓	✓	✓*	✓*
2 MEM WR	✓	✓	✓*	✓*
3 MEM R/W	✓	✓	✓*	✓*

\* Breakpoints A and B will both be of same type, such as MEM RD or MEM WR. Type may be specified for either A or B and will be in effect for both comparators.

#### 4-1.5 TRACE MEMORY

One of the most useful EM-186 features is its 254 bus cycle Trace Memory. The Trace Memory is organized as a ring buffer that records all target program activity. It operates in both real-time and single-step modes, and its contents remain in the correct sequence, regardless of whether the user operates the program wholly or partly in either of these two modes.

To review the Trace Memory contents, the EM-186 must be paused. The PAUSE mode is entered automatically when the program encounters a breakpoint, or it can be entered manually by depressing STEP. When the program enters PAUSE as a consequence of depressing the STEP Key, the Display shows the fetch cycle address and data for the last instruction recorded.

When a breakpoint event triggers PAUSE, the Display shows the cycle where the breakpoint was detected, and the user can easily review the program activities leading up to the event. Depressing DEC allows the user to examine the last 254 bus cycles of program activity prior to the breakpoint. Depressing INC allows the user to review forward up to the last cycle traced. Depressing STEP advances the target program past the breakpoint event, one instruction at a time. Depressing TRACE allows the user to return to Trace Memory again

after selecting another mode (i.e., MEM ADDR, etc.) and return the original program event or bus cycle to the display. The TRACE Key has no effect unless the program is already in PAUSE. STEP actually causes the emulator to execute another program instruction and this instruction is entered into the Trace Memory like all others.

The 6800/6802 machine instructions may have one or several bus cycles per instruction. The following two examples illustrate displayed Trace Memory contents, first after executing a simple instruction and then after a more complex one.

**EXAMPLE 1      TAB**

Cycle	Addr	Data	Fetch	RD	WR
1	4000	16	X	X	
2	4001	00		X	IREV CYC

Single bus cycle instruction: Move contents of A register to B register. Assume the instruction location is address  $4000_{16}$  in the target memory. The Trace Memory records a bus cycle with the address of  $4000_{16}$ , data of  $16_{16}$  and control bits indicating that it is a fetch operation and a read cycle.

**EXAMPLE 2      STX      \$7055**

Cycle	Addr	Data	Fetch	RD	WR
1	4000	FF	X	X	
2	4001	70		X	
3	4002	55		X	
4	7055	IREV	-	-	-
5	7055	34			X
6	7056	12			X

Six bus cycle instruction: Cycle one fetches op-code FF of the ST instruction located at address  $4000_{16}$ . Cycles two and three read high order and low order bytes ( $70_{16}$  and  $55_{16}$ ) of the 16-bit address located at  $4001_{16}$  and  $4002_{16}$ . Cycle four is an irrelevant cycle. Cycles five and six write the contents of the X register pair ( $34_{16}$  and  $12_{16}$ ). The Trace Memory records all bus cycles (except cycle 4) of the instruction. The address location, program data and opcode cycle are shown on the Display Panel for each bus cycle of the instruction. If the EM-186 had entered PAUSE and displayed Cycle 1 (the OP-CODE fetch), then the INC Key would be used to advance through the Trace Memory and observe the subsequent bus cycles.

Normally, the INC and DEC Keys move the trace index one cycle at a time. If the operator depresses and holds down the TRACE Key, however, the INC and DEC Keys will cause the next or previous fetch cycle to be displayed without stopping on other machine cycles.

The Trace Memory of the EM-186 may also be enabled to record irrelevant cycles, illegal instructions and nonVMA cycles by using the Code Functions E5 through E7. See Section 7-5.



**4-2 EXAMINATION AND  
ALTERATION OF MPU  
REGISTERS**

The 6800/6802 register contents may be examined by the operator, and if desired, overwritten with new data.

Register data is accessed for display by using the blue REG Keyswitch, followed by one of the hexadecimal keyswitches. This designates which register should be displayed. Table 4-4 shows the registers selected by the various keyswitches. Note that 0 through 7 do not correspond to actual 6800 registers. These keyswitches are used to set up parameters for the Built-In test routines, User Code Functions or software vectors. These Code Functions are described in later sections. (See Section 7 and Section 8).

Examples of readout and alteration of MPU registers:

**EXAMPLE:**



Accumulator B contents displayed on address display.



Accumulator B is accessed and then overwritten with data  $3F_{16}$ .



Stack Pointer is accessed and displayed on 16-bit address display.



Program Counter is accessed, and then the contents are overwritten with  $3C00_{16}$ .

All register contents are displayed on the Address Display. If the register has 16 bits, then all four of the digits of the display will be illuminated. If the register has eight bits, then the value is shown on the two loworder digits of the Address Display.

The Data Display is used for feedback to the operator of the register that has been selected. If the stack pointer is selected, the Data Display will show an 'E' since the E key is used to select that register. If Accumulator B is selected as in the example preceding, the Data Display will show a 'B'.

TABLE 4-4. Keyboard Designators  
(After REG is Pressed)

KEY	REGISTER	DESCRIPTION
0	IRQ *	Interrupt Request Vector
1	SWI *	Software Interrupt Vector
2	NMI *	Non-Maskable Interrupt Vector
3	RST *	Restart Vector
4	BEG *	BEGIN Address (for programmed tests)
5	END *	END Address (for programmed test)
6	ADDR *	Address (programmed test parameter)
7	DATA *	Data (programmed test parameter)
8	AB	Accumulators A and B
9	AB	Accumulators A and B
A	A	Accumulator A
B	B	Accumulator B
C	CC	Condition Code
D	PC	Program Counter
E	SP	Stack Pointer
F	X	Index Register

\* Not an actual 6800 or 6802 register.

### 4-3 EXAMINATION AND ALTERATION OF MEMORY LOCATIONS

Any memory location accessible to the emulated microprocessor may be accessed and displayed by the EM-186. If desired, new data may be written to the location.

#### EXAMPLE 1:


 Address  $431A_{16}$  is entered, and when EXAM is pressed, the EM-186 will read from address  $431A_{16}$  and display the data obtained.

If the operator wishes to review a group of sequential memory locations, this may be done by entering the initial address and examining that location as above; then examine successive locations by depressing INC.

#### EXAMPLE 2:


 Examine data at  $4300_{16}$ .


 Examine data at  $4301_{16}$

etc.

Examine data at successive locations, etc.

A memory location may be altered by entering an address, as shown above, then entering data using LOAD DATA and finally storing the data to the selected memory address using STORE.



**EXAMPLE 3:**



The above sequence writes the data  $55_{16}$  to memory address  $13FE_{16}$  in the target system.

Sequential locations may be quickly altered by incrementing the address after each data entry operation. For example, the following keystroke sequence enters a short program fragment into memory:

**EXAMPLE 4:**



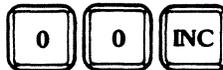
Enter initial address  $0800_{16}$ .



Enter data  $7E_{16}$  then store the data to  $0800_{16}$  and increment to  $0801_{16}$ .



Enter data  $08_{16}$  then store the data to  $0801_{16}$  and increment to  $0802_{16}$ .



Enter data  $00_{16}$  and store to  $0802_{16}$ ; increment to  $0803_{16}$ , etc.

The EM-186 does not require redundant keystrokes. The unit assumes that if the operator has entered new data while a particular memory address is accessed, then the operator wants to store that data before going to the next address.

In all of the above examples in which INC was used, DEC (decrement) could also have been used.

---

## **RAM OVERLAY**

---

**5-1 Overview**

**5-2 Installation**

**5-3 Controls**

**5-4 Uploading/Downloading**

---



### 5-2 INSTALLATION

To install a RAM Overlay option in your emulator (units without the two hex standoffs, as shown in Figure 5-2.1), the procedure is as follows:

1. Unplug the power cord from the back of the emulator.
2. Unplug the pod cables attached to the back of the emulator, noting the proper positioning.
3. Turn the emulator upside-down and place it on a soft surface to prevent scratching the top cover.
4. Remove the four top cover securing screws and the four rubber feet from the bottom of the emulator.
5. Remove the bottom cover.
6. Turn the emulator upright and carefully remove the topcover.
7. Remove the four display assembly securing screws, being careful not to scratch the display plex panel. Note the location of the spacers behind the plex panel.
8. Remove the display assembly.
9. Remove the four keyboard assembly securing screws.
10. Remove the keyboard assembly.
11. Unplug the four-contact Molex connector from the power supply regulator board, located next to the power switch.
12. Remove the two securing screws located on either side of the pod cable connector at the rear of the unit.
13. Slide the main logic assembly forward and out of the emulator frame. Note that it is necessary to install the new mounting bracket included in your 64K RAM kit onto the Main Logic Board.
14. Install the two hex standoffs supplied into the position shown in Figure 5-2.1. The standoffs will be secured into place by using two screws supplied in the installation kit. (Install the screws from the bottom side of the assembly).
15. Cut the lower half of the back panel decal to expose the cutouts as shown in Figure 5-1.1a for the 8K and 16K RAM Overlays only. For the 64K RAM Overlay, remove the old decal and replace it with the new one supplied in your 64K RAM Kit.
16. Plug the RAM Overlay connector into the main logic assembly.
17. Reassemble the emulator by reversing steps 1 through 13.
18. Connect the probe tip assembly to a known-good target system.

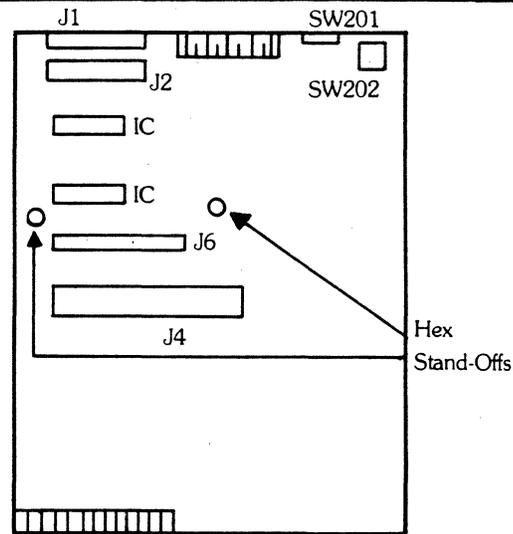


Figure 5-2.1 RAM Overlay Installation

If a RAM Overlay block is set up to respond to a range of addresses, say  $0000_{16}$  to  $0FFF_{16}$ , then target system memory in the same address range becomes inaccessible to the emulation processor. The memory block has "overlayed" the corresponding target system addresses. (See, however, the description of Code Function C5 for an exception to this characteristic of the emulator.)

The contents of the RAM Overlay are retained as long as power is applied to the emulator. It is possible to load the RAM Overlay with data, turn the enable switches off and retrieve the data at a later time. To retrieve data, turn the enable switches to either the READ or RD/WR position.

Each 4K byte block of memory for 8K and 16K RAM Overlays has an associated Address Range switch. The 64K RAM Overlay also has an associated control switch for each 4K block of memory. See Figure 5-3.1.

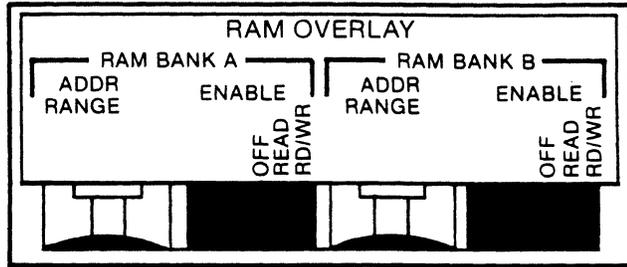
### 5-3 CONTROLS

The enable switches are three-position toggle switches that place the memory bank into one of three conditions:

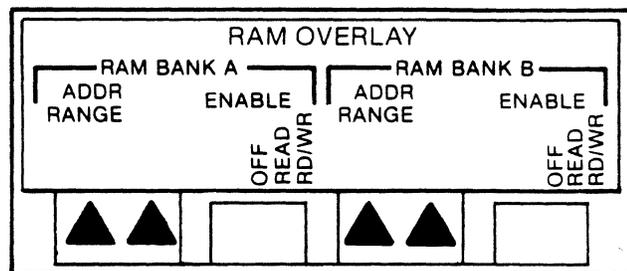
- **OFF**  
The memory bank is disabled and is effectively removed from the system.
- **Read**  
The memory bank is placed in a Read/Only configuration so that from the point of view of the target system the memory bank behaves like memory.
- **Read/Only**  
In this mode, it is not possible for the target system program to alter the contents of the memory. Note, however, that the emulator is still able to write to the memory bank from the keyboard or from a Code Function routine such as Code Function C3 (download).
- **Read/Write**  
The memory bank is placed in a Read/Write configuration. Both the target system and the emulator are able to read the memory and write new information to it.

# SECTION 5

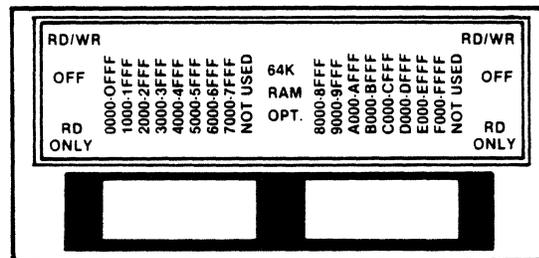
## RAM OVERLAY



a). 8K



b). 16K



c). 64K

Figure 5-3.1  
RAM Overlay Controls.

If a memory bank is disabled (toggle switch in the off position), the memory will nevertheless continue to retain data. The data will reappear in the target address space whenever the memory is again enabled.

The address range switches for the 8K and 16K RAM Overlays are 16-position rotary switches used to select the address range where the 4K memory blocks will reside in the target address space. Each of the 4K memory blocks can be moved to any of 16 positions, beginning at a 4K boundary. See Table 5-1.



**TABLE 5-1  
MEMORY BLOCK ADDRESS  
A and B**

SWITCH POSITION	MEMORY BLOCK ADDRESS	SWITCH POSITION	MEMORY BLOCK ADDRESS
0	0000 <sub>16</sub> -0FFF <sub>16</sub>	8	8000 <sub>16</sub> -8FFF <sub>16</sub>
1	1000 <sub>16</sub> -1FFF <sub>16</sub>	9	9000 <sub>16</sub> -9FFF <sub>16</sub>
2	2000 <sub>16</sub> -2FFF <sub>16</sub>	A	A000 <sub>16</sub> -AFFF <sub>16</sub>
3	3000 <sub>16</sub> -3FFF <sub>16</sub>	B	B000 <sub>16</sub> -BFFF <sub>16</sub>
4	4000 <sub>16</sub> -4FFF <sub>16</sub>	C	C000 <sub>16</sub> -CFFF <sub>16</sub>
5	5000 <sub>16</sub> -5FFF <sub>16</sub>	D	D000 <sub>16</sub> -DFFF <sub>16</sub>
6	6000 <sub>16</sub> -6FFF <sub>16</sub>	E	E000 <sub>16</sub> -EFFF <sub>16</sub>
7	7000 <sub>16</sub> -7FFF <sub>16</sub>	F	F000 <sub>16</sub> -FFFF <sub>16</sub>

Because of the large memory capacity, the card setup for the 64K RAM is slightly different than the 8K and the 16K RAM. The memory control switches consist of sixteen three-position switches. Eighteen switches are present, however two are not used. Each switch represents a 4K byte segment of memory. The switch in the up position enables the RD/WR memory; in the down position enables the RD only memory; and the center position disables the memory. The decal on the back of the EM denotes the switch positions and memory range.

**5-4 UPLOADING/  
DOWNLOADING**

Programs can be transferred to the RAM Overlay from the target system, the front panel EPROM socket or the RS-232C serial port on the auxillary connector. Programs in the RAM Overlay can be dumped to the RS-232C port. Both uploading and downloading is accomplished by enabling the RAM Overlay for read only or read/write, selecting the desired address range via the rotary switches, and then executing the appropriate Code Functions C1 through C6. A summary of these Code Functions is given below. (For more information see Sec. 7, Built-in Diagnostic Functions.) Note that the overlay block involved must be enabled; otherwise, these code funtions will involve the target system memory in place of the RAM Overlay.

**CODE C1 – LOAD RAM OVERLAY FROM FRONT PANEL PROM**

This Code Function transfers data from the front panel diagnostic EPROM to the Overlay RAM. To use this Code Function, first enter the starting and ending address values in the BEG and END registers, then start the routine.

**CODE C2 – VERIFY RAM OVERLAY AGAINST FRONT PANEL PROM**

Code C2 compares the front panel PROM with the address range specified by the user. The address range is loaded into the BEG and END register. If a non-verify occurs, the Diagnostic Emulator emits three beeps and pauses. The address and the data that failed to verify are shown on the display. By depressing and holding the EXAM keyswitch, the correct data will be displayed.

**CODE C3 – LOAD RAM OVERLAY FROM SERIAL LINK**

Code C3 transfers hex data from the RS-232C input to the RAM Overlay. To use this Code Function, connect the RS-232C input to the source of information, start the routine and then enable the source to download the appropriate data.

### **CODE C4 – DUMP RAM OVERLAY TO SERIAL LINK**

Code C4 transfers data from selected areas of RAM Overlay to the serial RS-232C output. To use the Code Function, first specify the address limits in the BEG and END registers, next prepare the receiving device to accept data, then start the routine.

### **CODE C5 – LOAD RAM OVERLAY FROM TARGET MEMORY**

Code C5 transfers data from selected areas of target memory space to the equivalent areas in Overlay Memory. The BEG and END registers are set to the range of addresses from which data is to be transferred.

### **CODE C6 – VERIFY RAM OVERLAY WITH TARGET MEMORY**

Code C6 compares data from selected areas of target program memory to the equivalent areas in Overlay Memory. The BEG and END registers are set to the desired target memory address range.

If a non-verify occurs, the Diagnostic Emulator emits three beeps and pauses. The address and the data that failed to verify are shown on the display. by depressing and holding the EXAM Keyswitch, the correct data will be displayed.



# SECTION 6

---

---

---

## DISASSEMBLY

---

**6-1 Overview**

**6-2 Format Definition**

---



## 6-1 OVERVIEW

The EM-186 Diagnostic Emulator may be configured with an enhanced firmware package that includes a disassembler. The disassembler firmware gives the EM-186 the ability to output the contents of the Trace Memory and emulation processor registers to the serial port; in this way a readable and attractive display may be created on a CRT or hardcopy terminal.

The disassembly firmware is disabled when the EM-186 is first powered up and must be enabled before use. The following procedure will make the EM-186 ready to operate with an ASCII terminal and disassembly firmware:

### 6-1.1 OPERATION PREPARATION PROCEDURES

1. Connect the terminal to the EM-186 using an appropriate cable. The minimum circuits that must be connected are:

Pin 1 - Protective Ground

Pin 2 - Serial Data Out

Pin 7 - Ground

Some RS232 terminals may also require the following connection:

Pin 20 - Data Terminal Ready

Take care that Pins 10, 11, 12, 13, 22, 23, 24 and 25 are not connected to incompatible circuits. See 9-1, Auxiliary Connector.

2. Set the Baud Rate Selector switches of the EM-186 and the terminal to compatible settings.
3. Check the setting of Option Switch 3. If Option Switch 3 is open (up), then the EM-186 will not output serial data unless the Clear-to-Send signal (Pin 5) is high. If the Clear-to-Send signal is not important in your application, set Option Switch 3 to the CLOSED (down) position and the EM-186 will output data on command regardless of the state of Pin 5.
4. Enable the disassembly firmware by executing the Code Function E1. The EM-186 is now ready to output to the terminal device. See Figure 6-2.1 for format example.

The disassembly firmware may be turned off by executing Code Function E0.

Operate the EM-186 in the normal manner. Any time that the EM-186 transfers from RUN to PAUSE, the disassembly firmware will format and dump a part of the contents of the Trace Memory to the terminal; normally 20 lines of output are produced, displaying about one-third of the Trace Memory contents. The last line output represents the last instruction executed and the firmware will then output the register display.

If the EM-186 is operated in single-step mode, the firmware will output the register display after every instruction.

The disassembly software is designed so that 20 lines of output are produced each time the emulator transfers from RUN to PAUSE; this amount of data provides approximately a full screen on most CRT terminals. If output of the entire contents of the Trace Memory is desired, execute Code Function D8. This Code Function will execute even if disassembly is not enabled, but the disassembly firmware option must be installed on the machine. All of the data in the Trace Memory will be formatted and output. Data output may be suspended for a moment by depressing the EXAM Key; when the Key is released, data output will continue.

**NOTE:** It is possible that the data recorded in Trace Memory does not represent actual machine execution of a program (for example, a block of data left by a memory diagnostic Code Function or a data transfer Code Function). In such a case, the disassembler will not format and output the data.

### 6-2 FORMAT DEFINITION

Figure 6-2.1 shows some lines from a printer connected to an EM-186. The various fields of the disassembly presentation are identified in the figure. All numbers that are output by the disassembler are in hexadecimal representation. Additional information about the fields of the display follows:

#### **Address**

The address of the first op-code byte of the instruction.

#### **Op-Code**

The operation code of the instruction.

#### **Operand**

The operand bytes of the instruction (if any).

#### **Op-Code Mnemonic**

The operation code of the instruction given in mnemonic form.

#### **Operand**

The operand field of the instruction in symbolic format, except that addresses and constants are given as hexadecimal numbers.

#### **Data Transfer**

Any data transfer operations that occur as a consequence of the instruction are shown here. The most common formats are:

AAAA > DD  
or AAAA < DD

The first format means that the processor wrote data 'DD' to address 'AAAA'. The second format means that the processor read data 'DD' from address 'AAAA'.



### **Data Transfer**

Some instructions transfer more than one byte of data. The second byte of a data transfer will be shown in this field. If there are more than two bytes transferred, the additional bytes are shown in fields 6 and 7 on the following lines. For example, all of the stack push cycles of the SWI instruction will be displayed in this manner. All of these transfers are easily seen from the display.

### **Condition Code Register**

The MPU condition code register is shown in this field. Each of the six characters in this field represent one of the condition code bits as follows:

- First - 'H' if half carry bit is true.
- Second - 'I' if interrupt mask bit is true.
- Third - 'N' if negative bit is true.
- Fourth - 'Z' if zero bit is true.
- Fifth - 'V' if overflow bit is true.
- Sixth - 'C' if carry bit is true.

If any of the condition code bits are not true, the letter is replaced by a period.

### **Accumulator A**

The content of Accumulator A after the execution of the instruction.

### **Accumulator B**

The content of Accumulator B following the execution of the instruction.

### **Index Register**

The content of the index register following the execution of the instruction.

### **Stack Pointer**

The content of the stack pointer following the execution of the instruction.

### **Program Counter**

The content of the program counter following the execution of the instruction.

# SECTION 6 DISASSEMBLY

Figure 6-2.1  
Code E1  
72-CHARACTER  
DISASSEMBLY  
FORMAT

Address	Op-Code	Operand	Op-Code Mnemonic	Operand	Data Transfer	Data Transfer	Condition Codes	Accumulator A	Accumulator B	Index Register	Stack Pointer	Program Counter
F895	24	03	BCC	F89A								
F89A	F6	D007	LDAB	D007	D007<50							
F89D	FB	D003	ADDB	D003	D003<08							
F8A0	F7	D007	STAB	D007	D007>58							
F8A3	F6	D006	LDAB	D006	D006<16							
F8A6	F9	D002	ADCB	D002	D002<08							
F8A9	F7	D006	STAB	D006	D006>1E							
F8AC	24	03	BCC	F8B1								
F8B1	81	F0	CMFA	#F0								
F8B3	26	CB	BNE	F880								
F880	FF	D002	STX	D002	D002>08	D003>08						
F883	F6	D005	LDAB	D005	D005<D7							
F886	FB	D001	ADDB	D001	D001<00							
F889	F7	D005	STAB	D005	D005>D7							
F88C	F6	D004	LDAB	D004	D004<DF							
F88F	F9	D000	ADCB	D000	D000<08							
F892	F7	D004	STAB	D004	D004>E7							
F895	24	03	BCC	F89A								
F89A	F6	D007	LDAB	D007	D007<58							
F89D	FB	D003	ADDB	D003	D003<08							
F8A0	F7	D007	STAB	D007	D007>60							
F8A3	F6	D006	LDAB	D006	D006<1E							
F8A6	F9	D002	ADCB	D002	D002<08							
F8A9	F7	D006	STAB	D006	D006>26							

HI.... 87 26 0808 D3FF F8AC

\* EM-186 outputs these values for the last instruction executed (and for every single step execution).



# SECTION 7

---

---

---

## BUILT-IN DIAGNOSTIC FUNCTIONS

---

- 7-1 Group A: Memory Tests**
  - 7-2 Group B: Oscilloscope Loops**
  - 7-3 Group C: Memory Load and Dump**
  - 7-4 Group D: Miscellaneous**
  - 7-5 Group E: Change Default Parameters**
  - 7-6 Group F: Internal Operations**
-



The Diagnostic Emulator contains Built-In test functions and utility routines designed to be convenient and useful for testing systems and verifying their proper operation. These test and utility routines have been named Code Functions because they are accessed by depressing the CODE keyswitch, followed by hexadecimal digits designating the routine desired. Table 7-1 lists all of the Code Functions programmed into the Diagnostic Emulator.

TABLE 7-1. CODE FUNCTIONS

**GROUP A: MEMORY TESTS**

A1	RAM TEST	(00/FF)	}	Repeating Tests
A2	RAM TEST	(Rotating 1s)		
A3	RAM TEST	(Addresses)		
A4	ALL RAM TESTS			
A5	ALL RAM TESTS		}	One Pass and Stop
A6	RAM TEST	(00/FF)		
A7	RAM TEST	(Rotating 1's)		
A8	RAM TEST	(Addresses)		

**GROUP B: OSCILLOSCOPE LOOPS**

B1	Repetitive Memory Read
B2	Repetitive Memory Write
B3	Repetitive Store/Examine Memory
B4	Repetitive Memory Write (Data/Data)
B5	Continuous Address Increment, 64K Range

**GROUP C: MEMORY LOAD AND DUMP**

C1	Load Target from Front Panel PROM
C2	Verify Target with Front Panel PROM
C3	Load Target from Serial Link (DOWNLOAD)
C4	Dump Target to Serial Link (UPLOAD)
C5	Load RAM Overlay from Target
C6	Verify RAM Overlay with Target
C7	Verify Target Against Serial Link
C8	Fill Target with Specified Data
C9	Verify Target with Specified Data
CB	Block Move Target System Data
CC	Dump Target to Serial Link in User Viewable Format
CF	Reload Target System Vectors

**GROUP D: MISCELLANEOUS**

D2	Display Clock Frequency
D3	Display PROM/ROM Signature
D4	Output 50 Nulls from Serial Link
D5	Call User Routine in Internal RAM at EC00 <sub>16</sub>
D6	Call User Routine in Internal RAM at EC03 <sub>16</sub>
D7	Clear Trace Memory
D8	Dump Entire Content of Trace Memory
D9	Halt MPU (for changing Front Panel PROM)
DA	Display Revision Number for Control PROM

DB	Display Revision Number for Disassembly PROM
DC	Calculate Branch Offset and Display
DD	Do Self-Test of Control PROM and Disassembly PROM
DE	Output (CR)(LF)(NUL)(NUL) from Serial Link
DF	Count Hours, Minutes and Seconds on Display

### GROUP E: CHANGE DEFAULT PARAMETERS

E0	Disable Disassembly (Default at Power On)
E1	Enable Disassembly, 72-Character Line (Also Enables Code E7)
E4	Hold Trace on Breakpoint
E5	Trace Non-VMA Cycles
E6	Trace Illegal Instructions (HCF'S: 3C, 3D, 9D, DD)
E7	Trace Irrelevant Cycles (SEL When DISASM)
E8	Turn Off E4
E9	Turn Off E5
EA	Turn Off E6
EB	Turn Off E7
EC	Turn Off All Four Options (E4-E7)

### GROUP F: INTROSPECTION MODE

F	Set Basic 'Introspection' Mode
F0	Set Introspection Mode and Initialize Emulator for
through	Debug of User Code Function Program
F9	

### 7-1 GROUP A: MEMORY TESTS

#### CODE A1 - 00/FF DATA TEST

The Code Function A1 memory test routine quickly determines whether all locations within a specified range can be set to 00<sub>16</sub> and FF<sub>16</sub>. The range tested is from the address specified by the BEG (begin) register through the address specified by the END register. The routine operates by setting the first location of the range to 00<sub>16</sub> and reading the location to see if a 00<sub>16</sub> is returned. Then the routine stores an FF<sub>16</sub> to this location and reads the location to see if an FF<sub>16</sub> is returned. Finally, the routine increments the address and tests the next location in the range. During the execution of this test, the address and data activity are visible on the displays and stored in the Trace Memory.

If a memory error is encountered, the routine emits 3 beeps and displays the address of the failure and the erroneous data read. At this time the operator has three options:

1. Depress EXAM to display the data the routine expected to read from the memory. Release the keyswitch to again display the bad data.
2. Depress INC to continue testing at the next address in the range. If additional problems are found, the program will stop again and any of the options listed may be taken.



3. Exit the test routine by using any of the mode keys (MEM, REG, RUN, etc.) or RESET.

After testing all locations in the specified range, the EM-186 emits one short beep and repeats the test. The RESET keyswitch is used to terminate the test at any time.

### **CODE A2 ROTATE 1s**

Code Function A2 memory test routine performs a test on all data bits in the range specified. The range tested is from the address contained in the BEG register through the address contained in the END register. The routine starts with the first location in the range and tests the location by writing and checking a bit, one bit at a time, in all of the positions of the word under test. The routine writes and checks by writing and reading the following data patterns:

<b>Binary Pattern</b>	<b>Hexadecimal</b>
0 0 0 0 0 0 0 1	01 <sub>16</sub>
0 0 0 0 0 0 1 0	02 <sub>16</sub>
0 0 0 0 0 1 0 0	04 <sub>16</sub>
0 0 0 0 1 0 0 0	08 <sub>16</sub>
0 0 0 1 0 0 0 0	10 <sub>16</sub>
0 0 1 0 0 0 0 0	20 <sub>16</sub>
0 1 0 0 0 0 0 0	40 <sub>16</sub>
1 0 0 0 0 0 0 0	80 <sub>16</sub>

After a location has been tested it is known that all bit positions in the location may be set and cleared independently of each other. The program then increments to the next sequential address in the range and proceeds to test in the same manner. If an error is detected, the test stops, the EM-186 emits the three beeps that signify an error, and the Display Panel shows the defective memory address and the bad data. At this point the operator has three options:

1. Depress EXAM to display the data the diagnostic routine expected to read (good data). Release EXAM to return the bad data to the display.
2. Depress INC to continue testing. If additional problems are found, the test stops and any of the options listed may be taken again.
3. To terminate the test, depress RESET, RUN, or any of the mode select keyswitches.

After testing all locations in the specified range, the EM-186 emits one short beep and repeats the test. The RESET keyswitch is used to terminate the test at any time.

### CODE A3 ADDRESS TEST

Code Function A3 memory test determines whether an address decoding failure exists in the memory system under test. It tests the memory range from the address contained in the BEG register to the address contained in the END register. The routine prepares for operation by clearing all locations in the range to  $00_{16}$ . Next, the first location is set to  $FF_{16}$  and then a check is made of all address related locations in the range to determine if any of them have been altered by the writing of the  $FF_{16}$ . After all locations in the range that are address-related to the first location have been checked, the program resets the first location to  $00_{16}$ . Then the next sequential location in the range is set to  $FF_{16}$ , and the address-related locations checked. The test proceeds until all locations in the specified range have been set to  $FF_{16}$ , and the respective address-related locations checked.

For the purposes of this test, an address is said to be related to a second address if it differs from it by only one bit (in any bit position). The test checks all possible address-related combinations as long as a generated address does not fall outside the specified range.

If an addressing error is found the test stops, the EM186 emits three beeps signifying an error, and the display shows the erroneous data and its address. At this point the operator has three options:

1. Depress EXAM to display the data the diagnostic routine expected to read. Release EXAM to return the erroneous data back to display.
2. Depress INC to continue testing. If additional problems are found, the test will stop and any of the options listed may be taken again.
3. To terminate the test depress RESET, or RUN or any of the mode selection keyswitches.

After all locations in the specified range have been tested, the EM-186 emits one short beep and repeats the test. To exit this code function at any time, depress RESET.



#### **CODE A4 - ALL TESTS AND REPEAT**

Code Function A4 executes the A1, A2, and A3 diagnostic functions in sequence, then emits a short beep and repeats. The test may be terminated by depressing RESET. In the event that an error is found, the operator may respond in any of the ways described for the individual diagnostic functions.

#### **CODE A5 - ALL TESTS AND STOP**

Code Function A5 executes the A1, A2 and A3 diagnostic functions in sequence, emits a short beep and stops. In the event an error is found, the operator may respond in any of the ways described for the individual diagnostic functions. When the test is complete, the displays will read CODE A5 and the Trace Memory will contain a record of the last 254 bus transfers.

#### **CODE A6 - 00 FF DATA TEST**

Code Function A6 is identical to the Code Function A1 except the function stops after a single pass through the test. When the test is complete, the displays will read CODE A6 and the Trace Memory will contain a record of the last 254 bus transfers.

#### **CODE A7 - ROTATE 1s**

Code Function A7 is identical to the Code Function A2 except that the function stops after a single pass through the test. When the test is complete, the displays will read CODE A7 and the Trace Memory will contain a record of the last 254 bus transfers.

#### **CODE A8 - ADDRESS TEST**

Code Function A8 is identical to the Code Function A3 except the function stops after a single pass through the test. When the test is complete, the displays will read CODE A8 and the Trace Memory will contain a record of the last 254 bus transfers.

### **7-2 GROUP B: OSCILLOSCOPE LOOPS**

The Oscilloscope Loop Functions are a group of functions that provide several types of repetitive stimuli to a target system. They provide repetitive waveforms in the target system hardware that may easily be examined at various circuit points with an oscilloscope or other test equipment. Many of the functions are also useful as stimulus routines for Signature Analysis testing.

### CODE B1 - REPETITIVE MEMORY READ

This function repetitively reads the single memory location addressed by the ADDR register. These address, data and RD signals are all shown on the Display. A high going pulse is output from the BKPT A pin (pin 12) of the Auxiliary Connector each time the memory location is read. Depress RESET to exit this function.

### CODE B2 - REPETITIVE MEMORY WRITE

This function repetitively writes the data contained in the DATA register to the single memory location addressed by the ADDR register. These address, data and WR signals are all shown on the Display. A high-going pulse is output from the BKPT A pin (pin 12) of the Auxiliary Connector each time the memory location is written. Depress RESET to exit this function.

### CODE B3 - REPETITIVE MEMORY WRITE/READ

This Code Function writes the data contained in the DATA register to the address specified by the ADDR register, then reads the same address; this process is repeated at a high-rate. A high-going pulse is output from the BKPT A pin (pin 12) of the Auxiliary Connector each time the memory address is accessed for either the write cycle or the read cycle. Depress RESET to exit this function.

### CODE B4 - REPETITIVE MEMORY WRITE (DATA/ $\overline{\text{DATA}}$ )

This function repetitively writes data to the address designated by the ADDR register. The data written is that contained in the DATA register except it is complemented every other time data is written. The address, data and WR signals are all displayed. A high-going pulse is output from the BKPT A pin (pin 12) of the Auxiliary Connector each time the address is accessed. Depress RESET to exit this function.

### CODE B5 - CONTINUOUS ADDRESS INCREMENT

This function places the EM-186 in a special mode in which it outputs successive addresses from  $0000_{16}$  to  $FFFF_{16}$  at a very high rate. Internal to the EM-186 this is accomplished by forcing a NOP instruction to the processor on every fetch cycle. Externally the EM-186 appears to be doing a fetch cycle at each address at the full speed of the processor (as determined by the clock frequency).

In this mode, the processor does **not** respond to target system WAIT commands.

The Continuous Address Increment function is used to check out address decoding networks in hardware systems and as a stimulus for signature analysis trouble shooting.



**7-3 GROUP C: MEMORY  
LOAD AND DUMP**

It is possible to obtain a sync pulse for triggering an oscilloscope or a signature analyzer from either the Breakpoint A or Breakpoint B output at the back Panel Auxiliary Connector; the output pulse occurs each time the processor reads from the breakpoint address. (The processor does not stop.)

Depress RESET to terminate the Continuous Address Increment mode.

**CODE C1 - LOAD TARGET FROM FRONT PANEL PROM**

The Code Function C1 transfers data from the Front Panel Diagnostic PROM to the target system. This routine requires the user to specify the destination address range in the target system by entering the first address of the range in register BEG and the last address of the range in register END. To use this Code Function, first enter the appropriate address values in the BEG and END registers, then start the routine. The routine will transfer bytes from the Front Panel PROM into the target address space with the first location in the PROM transferred to the first address of the specified range. After the transfer is complete, the Trace Memory contains a record of the last 254 cycles of the transfer.

**CODE C2 - VERIFY TARGET WITH FRONT PANEL PROM**

The Code Function C2 compares the Front Panel PROM with the address range specified by the user. The address range should be specified using the BEG and END registers in the same manner as described for C1.

**CODE C3 - LOAD TARGET FROM SERIAL LINK (DOWNLOAD)**

The Code Function C3 transfers hex data from the serial RS-232C input to the target system. The data to be entered must first be converted into the Motorola S format, which is an ASCII-hexadecimal format.

The destination address range is specified by the incoming data and need not be specified in the BEG and END registers. Furthermore, if the data is properly received, the BEG and END registers contain the low and high limits of the loaded data, regardless of the initial register settings. In addition, the limits will always be correct even if noncontiguous data is loaded.

To use this Code Function, connect the RS-232C input to the source of information, start this routine and then enable the source to 'download' the appropriate data. During the transfer, note the displays showing the data being loaded. If there are no errors, the end-of-file record completes the transfer and the displays contain CODE C3. The Trace Memory contains a record of the last 254 cycles of the transfer.

# SECTION 7

## BUILT-IN DIAGNOSTIC FUNCTIONS

**C3 ERROR CODES** - During the data transfer process, various types of errors can occur. If an overrun or a target memory write error is detected, the EM-186 stops transferring data, emits three beeps and displays error code 02 or 21 respectively. Since these two types of errors are considered 'fatal,' the transfer process cannot be resumed. If framing, non-hexadecimal, or sum-check errors are detected, the EM-186 will continue the transfer process without stopping. At the end-of-file, the front panel will display the error code of the highest priority error that occurred (see Table 7-2). The user can determine the number of each type of 'non fatal' error that occurred by executing the Code Function F and examining the following locations in the EM-186 internal address space:

Address	Error Type
EFB6	Non-Hexadecimal
EFB7	Sum-Check
EFB8	Framing

At each of these addresses, the number of errors will be given in hexadecimal. The C3 Code Function Error Codes are listed in Table 7-2.

Table 7-2. C3 Code Function Error Codes

### CODE PRIORITY DESCRIPTION

02	F*	Overrun Error. This error may occur if the processor is operated with an extremely low clock frequency while receiving data at high baud rates.
21	F*	Target Memory Write Error. If an attempt is made to load data to an area of memory containing no RAM or faulty RAM, this error occurs. This error is detected by doing a read-back-check of each location as it is stored, and recording both the write cycle and the read cycle in Trace Memory. After the error occurs register ADDR will contain the address of the faulty location while register DATA will contain the data the EM-186 attempted to store.
01	1	Framing Error. The serial data character is not properly framed by start and stop bits. This error may be caused by an incorrect setting of the baud rate selector or by noise on the transmission link.
11	2	NonHexadecimal Character Received. This error indicates that a hex character was expected at some point, but a non hex character was received.
12	3	SumCheck Error. In the Motorola S format each record contains an eight-bit checksum to ensure data integrity. If this sum is incorrect, this error code is given.

\*F = Fatal



#### **CODE C4 DUMP TARGET TO SERIAL LINK**

The Code Function C4 transfers data from a selected area of Target Memory to the serial RS232C output. The data being output is ASCII Hexadecimal and is compatible with the Motorola S format. The use of this function requires the user to specify the address range. The BEG register contains the starting address while the END register contains the address of the last location to be output. (See Sec. 9-4).

To use this Code Function, first specify the address limits, next prepare the receiving device to accept data, then start the transfer by executing CODE C4. During the transfer the display shows the address and data currently being transmitted. When transmission is completed, the displays show CODE C4 and the Trace Memory contains a record of the last 254 cycles.

The rate of transfer can be controlled by the receiving device. If enabled by the Option-Switch (with position 3 open), the CTS line (Clear-to-Send) can prevent output if held in the marking (negative) condition. In the spacing (positive) condition, output speed is determined by the baud rate selected.

Each record is followed by a carriage return, line feed and two null characters.

#### **CODE C5 - LOAD OVERLAY RAM FROM TARGET MEMORY**

The Code Function C5 transfers data from a selected area of target memory space to the equivalent area in Overlay Memory. To use this Code Function the Overlay memory must first be located at the proper address by rotating the thumb-wheel switch. The Overlay is then enabled by setting the selector switch to the appropriate position. Then the BEG and END registers are set to the range of addresses over which data is to be transferred. The last step is to call the Code Function to execute the transfer. While executing, the displays show the addresses and data. When data transfer is completed, the displays show CODE C5 and the Trace Memory contains a record of the last 254 cycles of the data transfer. (See Sec. 9-4.)

If a non-verify occurs during the transfer, the Diagnostic Emulator emits three beeps and temporarily halts the transfer. The error may be skipped and the transfer resumed by depressing INC, or the operation may be aborted by depressing a mode select keyswitch, such as CODE. While the operation is halted, the address and the data that failed to verify is shown on the display. By depressing and holding the EXAM keyswitch, the correct target data may be displayed.

#### **CODE C6 - VERIFY RAM OVERLAY WITH TARGET MEMORY**

The Code Function C6 compares data from a selected area of Target Memory to the equivalent area in Overlay Memory.

For information on the operation of this function, see Code C5.

### **CODE C7 - VERIFY TARGET WITH SERIAL LINK**

The Code Function C7 is nearly identical to the C3 Code Function. It differs in two respects:

1. Data is not stored to target memory but only verified.
2. A nonverify results in Error 22 and the compare operation is aborted. Register ADDR will contain the address of the noncompare while register DATA will contain the data that was supposed to be in the target memory location.

### **CODE C8 - FILL MEMORY WITH DATA**

The Code Function C8 is used to fill a block of target memory or RAM Overlay with the same data, usually all one's (FF) or all zeros. To use this Code Function, set the BEG and END registers to the range of target memory or RAM Overlay to be filled, load the DATA Register with the data to be stored then execute CODE C8. The Display shows the transfer as it takes place. After transfer is completed the display shows CODE C8 and the trace contains a record of the last 254 cycles of the transfer.

If a location fails to store the correct data, the Diagnostic Emulator emits three beeps and temporarily halts the fill operation. The error may be skipped and the transfer resumed by depressing INC, or aborted by depressing a mode select keyswitch such as TRACE. While the operation is halted the address and the data that failed to verify are shown on the Display. By depressing and holding EXAM the correct data (which was in DATA) may be displayed.

### **CODE C9 - VERIFY MEMORY WITH DATA**

The Code Function C9 compares a block of target memory or RAM Overlay with the byte in register DATA. See the explanation of CODE C8 for the operation of the function.

### **CODE CB - BLOCK MOVE**

The Code Function CB is used to move a block of data residing in the target system to a new location in target system RAM. Define the block of data to be moved by entering the address of the first byte of the block in the BEG register and the address of the last byte of the block in the END register. Enter the address of the first byte of the destination block in the ADDR register. Execute Code Function CB to move the data.

This routine is able to move the block of data to a higher address or to a lower address. In addition, the blocks may overlap in any manner and moved without loss of data; for example, a block of 2K bytes could be moved up or down by fifteen positions.

The entire destination block must be in writeable memory.

**CODE CC - DATA OUTPUT TO SERIAL PORT IN HEX AND  
ASCII FORMAT (Disassembly Firmware Required)**

This Code Function provides a formatted dump of a block of memory to the serial port. The memory block is defined by the addresses contained in the BEG and END registers. When the function is executed, data will be output from the serial port in the format shown in Figure 7-3.1.

Figure 7-3.1. Memory Dump Format

ADDRESS <sub>16</sub>	DATA <sub>16</sub>	ASCII
0000	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F	.....
0010	10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F	.....
0020	20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F	!"#\$%&'()*+,-./
0030	30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F	0123456789:;<=>?
0040	40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F	@ABCDEFGHIJKLMNO
0050	50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F	PQRSTUVWXYZ[\]^_`
0060	60 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F	'abcdefghijklmnop
0070	70 71 72 73 74 75 76 77 78 79 7A 7B 7C 7D 7E 7F	qrstuvwxyz{ }~.
0080	80 81 82 83 84 85 86 87 88 89 8A 8B 8C 8D 8E 8F	.....
0090	90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F	.....
00A0	A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 AA AB AC AD AE AF	.....
00B0	B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB BC BD BE BF	.....
00C0	C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF	.....
00D0	D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 DA DB DC DD DE DF	.....
00E0	E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE EF	.....
00F0	F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FO	.....
0100	07 07 DD 77 00 07 DD 77 01 07 DD 77 02 07 DD 77	...w...m...w...n
0110	03 07 DD 77 04 07 DD 77 05 07 DD 77 06 07 DD 77	...w...m...w...n
0120	07 C9 21 00 10 DD 21 00 30 0E 00 1E 00 41 0C 78	..!...!.0...A.x
0130	A9 A1 28 1F 47 AB 5F 16 86 78 A3 28 02 16 C6 78	..(.G...x.(...x
0140	06 07 0F 38 02 10 FB 36 DD 23 36 CB 23 70 23 72	...8...6.#6.#F#r
0150	23 18 DA 36 E9 21 74 01 DD CB 07 86 DD CB 06 86	#..6.!t.....
0160	DD CB 05 86 DD CB 04 86 DD CB 03 86 DD CB 02 86	.....

**NOTE:** Memory data formatted into lines of 16 Bytes with the address of the first Byte at the left margin.

### 7-4 GROUP D: MISCELLANEOUS

#### CODE CF - RELOAD TARGET SYSTEM VECTORS

This code function will read the vectors of the target system ROM and rewrite them in the soft vector RAM. This provides a quick method of resetting the soft vectors to the values contained in the ROM. (See Section 9-2.)

#### CODE D2 - DISPLAY CLOCK FREQUENCY

This Code Function is a routine that determines the clock frequency of the emulation processor by comparing the instruction execution rate of the processor with the EM-186 internal 1.2 KHz reference frequency. The internal reference frequency is derived from the crystal controlled UART clock. The frequency is displayed on the ADDRESS display and is given in kilohertz. For example, a 6800 operating with a 1.0 MHz clock will display 1000 (kilohertz) on the ADDRESS displays when CODE D2 is executed. The result is accurate to about  $\pm .01\%$  (the accuracy of the UART crystal).

#### CODE D3 - DISPLAY PROM/ROM SIGNATURE

The purpose of this Code Function is to provide a convenient way of verifying that all bits in a PROM or ROM are correct. The routine operates by reading each 8-bit byte in a specified range and shifting the bits into a firmware implemented feedback shift register. By this means, the routine calculates a 16-bit check value that is displayed as a 4-digit hexadecimal signature on the ADDRESS display. This signature has a very high probability (.9998) of being unique for any given bit pattern in a ROM.

A PROM or ROM signature is obtained by setting the first address of the ROM in the BEG register and the last address of the ROM in the END register; then execute the routine. The Code Function routine will calculate and display the ROM signature. If the correct signature has been obtained previously with a known good ROM, then the ROM under test is good if it has the same signature.

There is a technique that may be used to create PROMs or ROMs whose signatures are zero. For the method to work, the first two locations of the ROMs must be unused. Proceed as follows:

1. Program a PROM with the desired information, making sure that the first two bytes are zeros.
2. Determine the signature of the PROM using the CODE D3 function.
3. Program a new PROM with the desired information, but replace the first bytes, which previously were zero, with the bit pattern of the signature obtained in Step 2.



Now, while the signature of the new PROM is being calculated, the routine will arrive at a point just prior to processing the first two bytes of the PROM and at that time, the shift register will contain the signature of the PROM as calculated in Step 2; entry of the first two bytes, containing the same bit pattern as that already in the shift register results in the shift register reaching a final value of zero when computation is complete. Thus, the PROM will have a signature of 0000.

#### **CODE D4 - OUTPUT 50 NULS TO SERIAL PORT**

This Code Function outputs 50 nul characters (00<sub>16</sub>) to the serial port for the purpose of providing leader or trailer for users using punched paper tape as a data storage media. There are no parameters for this Code Function.

#### **CODE D5 - CALL USER ROUTINE IN INTERNAL RAM AT EC00<sub>16</sub>**

#### **CODE D6 - CALL USER ROUTINE IN INTERNAL RAM AT EC03<sub>16</sub>**

These two Code Functions provide a means for the user to transfer control to routines that have been entered into the EM-186 internal scratch pad RAM for various reasons. To make use of this feature, the user must understand the requirements of the programs that run in the EM-186 internal environment. See Section 8 User Implemented Code Functions.

#### **CODE D7 - CLEAR TRACE MEMORY**

The EM-186 Trace Memory is cleared when power is applied as part of the power onreset operations. Code Function D7 is used to clear the Trace Memory at any other time. This routine does not use any parameters.

#### **CODE D8 - DISASSEMBLE AND OUTPUT ENTIRE CONTENT OF TRACE MEMORY (Disassembler Firmware Required)**

This Code Function outputs the entire content of the Trace Memory to the serial port in the standard disassembler format. This routine may be called even if the regular disassembly feature of the EM-186 is disabled. Data output may be suspended for a moment by depressing the EXAM key; when the key is released, data output will continue. See Section 6 - Disassembly.

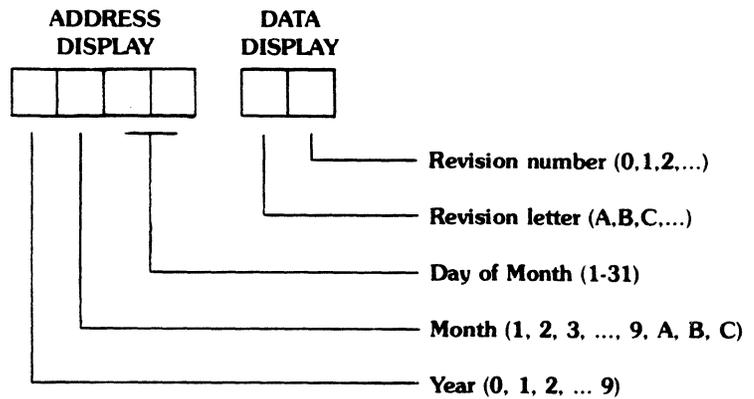
#### **CODE D9 - HALT MPU**

Code Function D9 causes the MPU to execute a HALT instruction, thereby halting the MPU. It is recommended that the MPU be halted any time that an EPROM is inserted into or removed from the Front Panel Diagnostic PROM Socket to avoid the possibility of crashing the internal control program of the EM-186. After the MPU has been halted, RESET must be used to resume normal operation. There are no parameters for this function.

### CODE DA - DISPLAY REVISION NUMBER OF CONTROL PROM

### CODE DB - DISPLAY REVISION NUMBER OF DISASSEMBLY PROM

These two Code Functions display the date and revision information of the control PROM software and the disassembly PROM software respectively. The format is as follows:



### CODE DC - CALCULATE BRANCH OFFSETS

Code Function DC is intended to simplify the task of calculating branch offsets for 6800 relative branch instructions. To use this routine, enter the address of the destination of the branch instruction in the BEG register; enter the address of the byte following the branch instruction in the ADDR register. Execute the function and the Address Display will show two digits that are the required branch offset. If the required offset is too large to be reached by branch instructions, then four digits will be displayed showing the offset.

### CODE DD - SELF TEST OF INTERNAL PROM DATA

Code Function DD is used to perform a check of the data in the internal PROMS—the Control PROM and the Optional Disassembly PROM. When this function is called, the data display will show 01 while the first 4K (Control PROM) is being tested and 02 while the second 4K (Disassembly PROM) is being tested.

A failure of this test will result in three beeps and the display will show EC 31 if the Control PROM failed or EC 32 if the Disassembly PROM failed.



**7-5 GROUP E: CHANGE  
DEFAULT PARAMETERS**

**CODE DE - OUTPUT LINE ENDING SEQUENCE TO SERIAL PORT**

This Code Function outputs the line ending sequence to the serial port consisting of a carriage return, a line feed, and two null characters. The routine is used to obtain a new line on a CRT or other ASCII display.

**CODE DF - DISPLAY HOURS, MINUTES, SECONDS**

**(Disassembly Firmware Required)**

Code Function DF places the EM-186 in a clock mode that counts hours, minutes and seconds on the Address and Data displays. To set the initial display, enter the desired hours and minutes into the ADDR register and the desired seconds into the DATA register. Then execute the function to start the clock. If the initial values are set to zero, then the clock will indicate elapsed time from 0000 00 to 1259 59 (13 hours).

**CODE E0 - DISABLE DISASSEMBLY (DEFAULT)**

Code Function E0 is used to disable the disassembly software if it is in operation. See Section 6 - Disassembly.

**CODE E1 - ENABLE DISASSEMBLY**

Code Function E1 enables the disassembly firmware and configures the firmware to output 72-character lines with one line of register display. This Code Function also enables the Code E7 which traces the irrelevant cycles. See Section 6.

**CODE E4 - HOLD TRACE ON BREAKPOINT**

Code Function E4 allows you to freeze the Trace Memory on breakpoint while continuing to execute your program. Option switch No. 2 on the EM-186 back panel must be in the OFF (up) position.

**CODE E5 - TRACE NON-VMA CYCLES**

Code Function E5 allows the Trace Memory to capture non-VMA cycles.

**CODE E6 - TRACE ILLEGAL INSTRUCTIONS**

Code Function E6 allows the Trace Memory of the EM-186 to capture illegal instructions.

**CODE E7 - TRACE IRRELEVANT CYCLES**

Code Function E7 allows the Trace Memory of the EM-186 to capture irrelevant cycles.

### 7-6 GROUP F: INTERNAL OPERATIONS

**CODE E8 - DISABLE CODE E4**

**CODE E9 - DISABLE CODE E5**

**CODE EA - DISABLE CODE E6**

**CODE EB - DISABLE CODE E7**

**CODE EC - DISABLE CODES E4, E5, E6, E7**

### **CODE F - SET INTROSPECTION MODE**

Execution of this Code Function sets the EM-186 so that its own internal address space becomes the "target system". After execution of the CODE F function, memory examine and store operations will be directed to the EM-186 internal address space; programs internal to the EM-186 may be executed in single-step mode and other internal operations performed. See Section 8 - User Implemented Code Functions.

### **CODE F0, F1, ... F9**

Code Functions F0 through F9 are used to set up the EM-186 to debug user programs residing in the front panel Diagnostic PROM Socket. These functions each set the emulator into "introspection" mode so that the internal address space is accessible, set the stack pointer to  $EEFF_{16}$  (the top of the internal user RAM area) and set the program counter to the starting address of the respective user programmed Code Function. The EM-186 is then ready to execute the user's program in singlestep mode or at full speed; breakpoints may be set and registers examined, and other normal debugging activities carried out (see Section 8-3, Entry to User Code Functions).



# SECTION 8

---

---

---

## USER IMPLEMENTED CODE FUNCTIONS

---

- 8-1 Overview**
  - 8-2 Internal Environment**
  - 8-3 Entry to User Code Functions**
  - 8-4 Introspection Mode**
  - 8-5 Getting To and From the Target System**
  - 8-6 User Accessible Subroutines**
  - 8-7 Interrupts**
  - 8-8 Code Function Examples**
-



## 8-1 OVERVIEW

The EM-186 Diagnostic Emulator has a low-insertion-force socket on the front panel that is designed to accept EPROMs similar to the Intel 2716 or 2732 devices. This front panel socket is called the Diagnostic PROM Socket. The purpose of the Diagnostic PROM Socket is to provide a means whereby the user may insert EPROMs programmed with his own diagnostic routines and execute them in a convenient manner from the EM-186 Keyboard. These user routines may perform almost any imaginable function. In most cases, the user will probably write special test or diagnostic routines to help test portions of the target system for which no Built-In Code Functions are provided. This discussion provides a view of the internal environment of the EM-186 from the programmer's perspective and is intended to provide the information needed by the user to write and debug his own Code Functions.

The programmer is already familiar with the environment of his own target system. He knows there is a 64K byte address space called the Memory Address Space and within this address space are various blocks of ROM, RAM or data registers.

The EM-186 also has an internal address space with its own ROM, RAM, and I/O. The EM-186 control program and Built-In Code Functions reside in this address space. Any EPROM plugged into the Diagnostic PROM Socket also appears in this internal address space. It is thus possible for User Code Functions to access all Diagnostic Emulator facilities and to function exactly as if they were factory-programmed.

However, the Code Function programs executing within the internal address space do not have direct access to the user's system target address space. If it is necessary for a Code Function to read or write to the external target system, it must do so in cooperation with the Diagnostic Emulator hardware circuits. Consequently, a rigidly defined routine must be executed to perform read or write operations to the target system. The Built-In Code Functions, as well as the EXAMINE and STORE routines, use EM-186 control program subroutines, and the user may also use these same subroutines to read and write to the target program address space.

# SECTION 8

## USER IMPLEMENTED CODE FUNCTIONS

### 8-2 INTERNAL ENVIRONMENT

The internal environment of the EM-186 contains ROM, RAM and I/O. The I/O devices of the EM-186 are memory mapped. Figure 8-2.1 shows an overview of the EM-186 internal address space.

Figure 8-2.1. EM186 Internal Memory Map

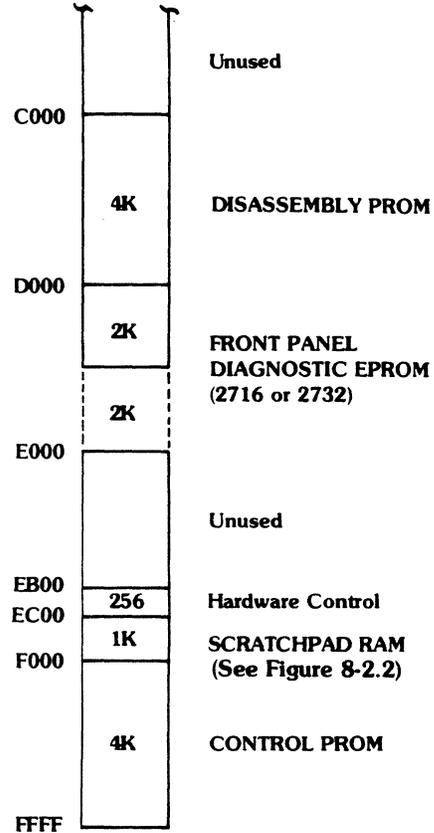




Figure 8-2.2. Map of Internal Scratchpad RAM

ADDRESS					
EC00	SCRAM			<b>User Written Programs and Scratch Area</b>	<b>(512 Bytes)</b>
EE00				<b>User Stack Area</b>	<b>(256 Bytes)</b>
EF00				<b>Firmware Stack Area</b>	<b>(80 Bytes)</b>
EF50				<b>Firmware Scratchpad</b>	<b>(144 Bytes)</b>
EFE0 EFE7				<b>Hardware Control Registers</b>	<b>(8 Bytes)</b>
EFE8	RGA	8	<b>Target Register A</b>	<b>(1 Byte)</b>	
EFE9	RGB	8	<b>Target Register B</b>	<b>(1 Byte)</b>	
EFEA	RGCC	8	<b>Target Condition Codes</b>	<b>(1 Byte)</b>	
EFEB	RGX	16	<b>Target Index Register</b>	<b>(2 Bytes)</b>	
EFED	RGSP	16	<b>Target Stack Pointer</b>	<b>(2 Bytes)</b>	
EFEF	RGPC	16	<b>Target Program Counter</b>	<b>(2 Bytes)</b>	
EFF1	RBEG	16	<b>Begin Pointer</b>	<b>(2 Bytes)</b>	
EFF3	REND	16	<b>End Pointer</b>	<b>(2 Bytes)</b>	
EFF5	RADR	16	<b>Current Address of Test</b>	<b>(2 Bytes)</b>	
EFF7	RDTA	8	<b>Current Data</b>	<b>(1 Byte)</b>	
EFF8	RIRQ	16	<b>IRQ Vector</b>	<b>(2 Bytes)</b>	
EFFA	RSWI	16	<b>SWI Vector</b>	<b>(2 Bytes)</b>	
EFFC	RNMI	16	<b>NMI Vector</b>	<b>(2 Bytes)</b>	
EFFE	RRST	16	<b>Restart Vector</b>	<b>(2 Bytes)</b>	
<b>TOTAL:</b>				<b>1024 Bytes</b>	

### 8-2.1 ROM

The EM-186 has two sockets, located on the Keyboard circuit card, that accept EPROMs or ROMs that contain the control program for the unit. The circuit board connections are normally set up for EPROMs or ROMs having the Intel 2732 pinout; a jumper modification of the board allows use of the 2K byte 2716 as well.

### 8-2.2 FRONT PANEL EPROM SOCKET

The EM-186 Front Panel EPROM Socket also accepts EPROMs of the 2716 or 2732 variety. A small switch located in the center of the socket selects the appropriate connections for either the 2K byte or 4K byte EPROM types. In the internal address space, the EPROM plugged into the Front Panel Socket will appear in the address range of  $D000_{16}$  to  $D7FF_{16}$  (2716) or  $D000_{16}$  to  $DFFF_{16}$  (2732). It is not possible to have this EPROM appear in the external (target) address space. See Figure 8-2.1.

### 8-2.3 SCRATCHPAD RAM

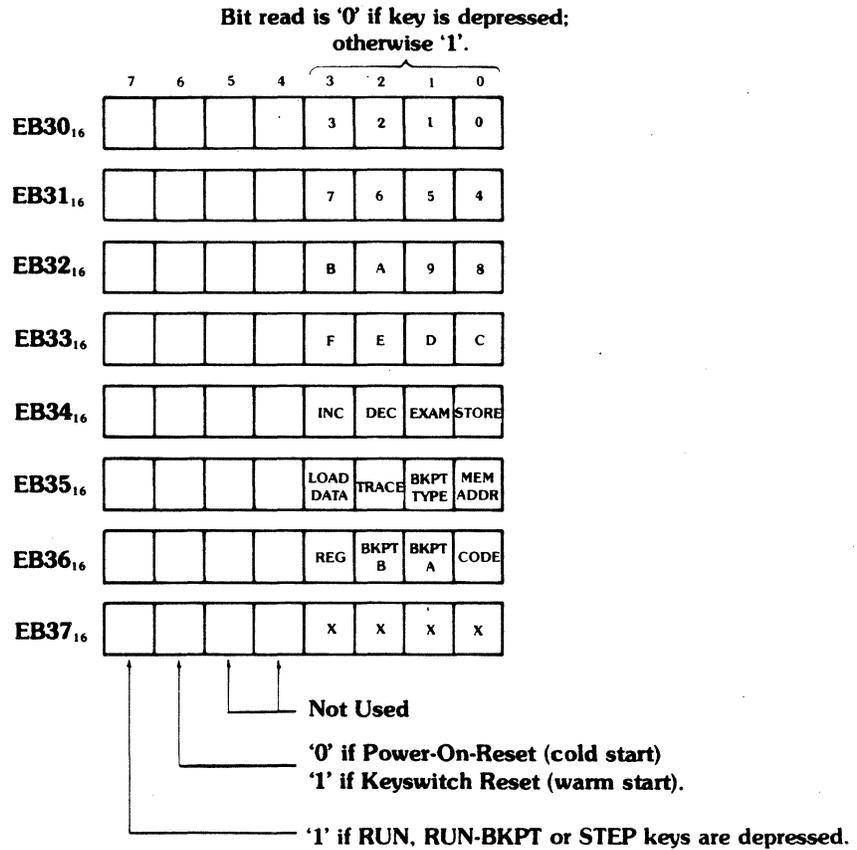
The EM-186 also contains a small amount of Scratchpad RAM in the internal address space that is used by the control program in keeping track of the status of the emulator and the emulation processor. The Scratchpad RAM resides in the internal address space at addresses  $EC00_{16}$  to  $FFFF_{16}$ . Figure 8-2.2 shows a detail of the Scratchpad RAM. The first 768 bytes of the RAM ( $EC00_{16}$  to  $EEFF_{16}$ ) are available to user-written Code Function programs.

The Scratchpad RAM also contains the area where the processor registers are saved each time the emulation processor pauses. User implemented programs may obtain these register values or even alter them if desired. The user should carefully avoid altering any of the data contained in the firmware stack area or firmware scratchpad locations to avoid crashing the control program.

### 8-2.4 I/O DEVICES

**KEYBOARD:** The state of the Keyboard Keyswitches may be read by the processor at a series of eight addresses from  $EB30_{16}$  through  $EB37_{16}$ . Four Keyswitches may be read at each of the input addresses as shown in Figure 8-2.3. A key depression causes the corresponding bit to go low as seen in the input data. For example, if Key 9 is depressed, bit 1 of location  $EB32_{16}$  will be low. Bits 4, 5, 6 and 7 of all eight of the input ports see the same data. Bit 4 and 5 in all locations are unused and tied high in the EM-186. Bit 6 will be low if the most recent system reset was caused by the power-on-reset circuitry; bit 6 will be high if the most recent system reset was caused by the RESET Key or a reset command from the target system. Bit 7 will be high if any of the following keys are depressed: RUN, RUNBKPT or STEP.

Figure 8-2.3. Keyboard Input Locations



The user who decides to write software to directly read the Keyboard must be aware that there is no key debouncing or other processing of the key closure done by the hardware. Consequently, it is necessary to provide the keystroke debouncing, repeating key features or other special processing in the software that scans the Keyboard. There is a Keyboard scan routine already in the EM-186 which may be accessed by the user that provides the most commonly needed features. See Section 8-6.

**SERIAL INPUT/OUTPUT PORT:** The EM-186 Diagnostic Emulator contains circuitry that implements a full-duplex (two-way) serial Input/Output port that conforms to RS-232C requirements. The baud-rate, parity and character length of the data transmitted and received is set up by hardware switches. (See Sections 9-2 and 9-3). The nature and format of data transmitted is under the control of software. The software is able to send data to the serial output circuits, read data from the serial input circuits and test the status of the serial port circuitry via three ports as shown in Figure 8-2.4. Data is transferred to and from the serial port by means of a Universal Asynchronous Receiver-Transmitter (UART).

# SECTION 8

## USER IMPLEMENTED CODE FUNCTIONS

Data to be output through the serial port is written to the UART data write address. The data enters the UART transmit buffer register, and then enters the transmit shift register where it is shifted out in serial form bit by bit. New data may be written to the transmit buffer register as soon as the previous data has entered the transmit shift register and before it has completed the process of shifting out.

The UART Status Register (Figure 8-2.4) contains two bits which inform the software of the status of the transmitter registers as follows:

Bit 1, Transmit Buffer Empty, will be read as a '1' when the transmit buffer register may be loaded with another character. A '0' means that the transmit data register contains data that has not yet been moved into the transmit shift register.

Bit 2, End of Character, will go to '1' at the time that a character has shifted out of the transmit shift register. If there is another character waiting in the transmit buffer register, then bit 2 will immediately go to '0' as the new character enters the shift register to be transmitted.

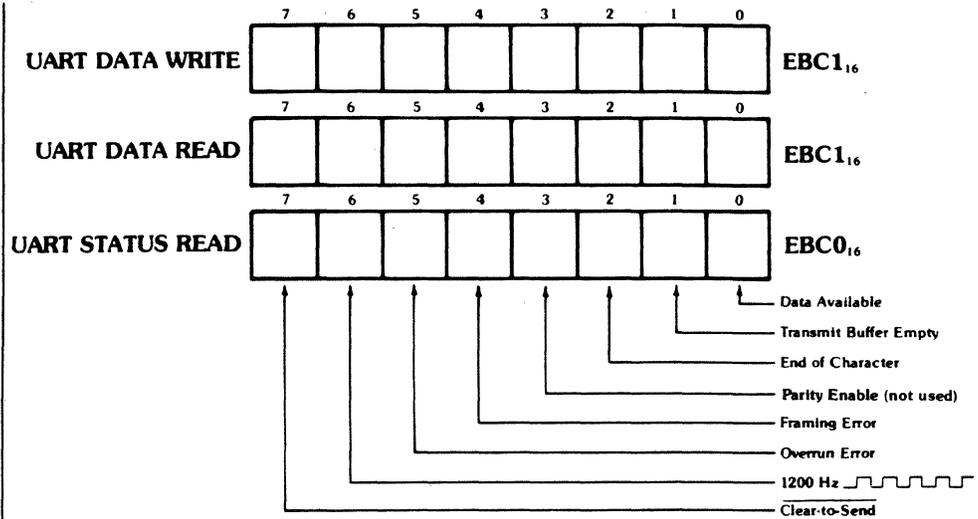
Data received by the EM-186 through the serial port is entered into the UART receiver shift register. When an entire character has been received, it is transferred to the receiver holding register and is then available to the software by reading the data at the UART Data Read address. Several status bits in the UART Status Register (Figure 8-2.4) give information about the received data as follows:

Bit 0, Data Available, goes to '1' when an entire character has been received and transferred to the receiver holding register. When the software reads the UART Data Read location, this bit is cleared to '0'.

Bit 4, Framing Error, goes to '1' if the received character has no stop bit at the expected location. This usually means that the transmitting device is sending characters of different length than the EM-186 is set up to receive. Noise may also cause this error.

Bit 5, Overrun Error, goes to '1' if a previously received character in the receiver holding register is not read by the CPU before another character is received and transferred into the holding register.

Figure 8-2.4. Serial Port Data and Status Locations



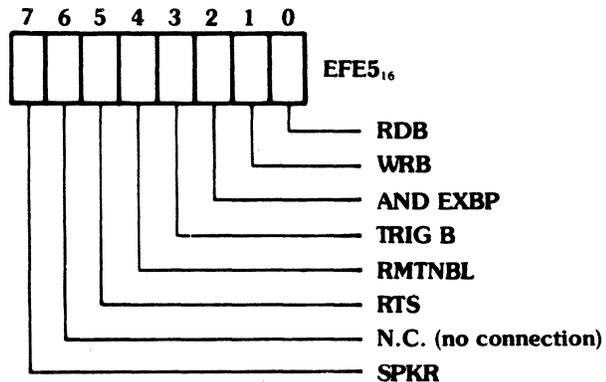
The clear-to-send input (Auxiliary Connector, Pin 5) is visible to the software as Bit 7 of the UART status register. This bit is '0' if clear-to-send is true (high); if clear-to-send is low or disconnected, this bit is '1'. (Note, however, that this bit may be forced to the '0' state by setting Option Switch 3 closed. See Section 9-2.)

Two other bits share the UART Status Register, but are not directly involved in the communications functions.

Bit 3, Parity Enable, is not used in EM-186.

Bit 6, 1200 Hz, is a 1200 Hz square wave that is derived from the bit rate-generator crystal oscillator.

Another status signal associated with the communications interface is the Request-to-Send (RTS) output on Pin 4 of the auxiliary connector. The state of this signal is visible to user written software as Bit 5 of the BKPT mode register located at address EFE5<sub>16</sub> as shown below:



# SECTION 8

## USER IMPLEMENTED CODE FUNCTIONS

Writing a '0' to Bit 5 of this register will set the RTS signal to its negative (OFF) state; writing a '1' sets RTS positive (ON).

When writing to Bit 5, it may be desirable to mask the other bits so that their status does not change. One way to accomplish this is shown in the simple program listing below:

### **:Turn RTS ON**

```
LDAA  $EFE5      ; Load register image
ORAA  #$20       ; Turn bit 5 ON
                    mask bits 0-4, 6, 7
STAA  $EFE5      ; Rewrite to register
```

### **:Turn RTS OFF**

```
LDAA  $EFE5      ; Load register image
ANDA  #$CF       ; Turn bit 5 OFF
                    mask bits 0-4, 6, 7
STAA  $EFE5      ; Rewrite to register
```

**HEXADECIMAL DISPLAYS AND TRACE MEMORY:** The EM-186 Trace Memory is a 254-word by 32-bit memory whose primary function is to record each bus cycle that occurs to the target system. At any given time, a single word of the Trace Memory is selected by an 8-bit register called the "XADDR" (trace index address) register. If, for example, the XADDR register contains a 43, then the next bus cycle will increment the register to 44 and write the data to 44. If the Emulation Processor were executing a target program, each bus cycle would increment the XADDR and write the data to that location. When XADDR reaches its maximum value of  $FF_{16}$  and is again incremented, it overflows to  $00_{16}$  so that the first location of the memory effectively follows the last location. Thus the Trace Memory may be viewed as a ring memory in which each additional bus cycle may be entered in the next position around the ring. Once the Trace Memory is full, each additional bus cycle simply overwrites the oldest bus cycle in the memory.

The Address and Data hexadecimal displays and the eight discrete Machine Cycle indicators are wired directly to the Trace Memory circuitry so that the current Trace Memory word (the word designated by the XADDR register) is always displayed unless the displays are explicitly blanked.



When the EM-186 is in the PAUSE mode, the internal control program has access to the Trace Memory and the displays by means of a set of five ports. See Figure 8-2.5. The five ports are as follows:

**XADDR (EB10<sub>1,6</sub>)** This port gives access to the Trace Index Address register. The control program may read this location to obtain the current value of the XADDR register, and may store new values in the register. Storing a new value in XADDR will change the current trace word that is accessed and displayed on the EM186 display panel.

**TDATA (EB50<sub>1,6</sub>)** This port gives access to the eight-bit wide portion of the current Trace Memory word that records the data bus signals of each machine cycle. The control program may read this location to obtain the data portion of the current trace word, or may store new data to the data portion of the trace word.

**TADDL (EB51<sub>1,6</sub>)** This port gives access to the eight-bit portion of the current trace word that records the low order eight bits of the address bus of each machine cycle. The control program may read or write this location.

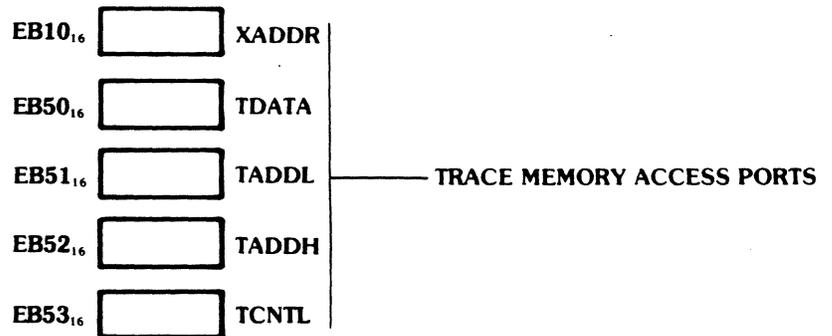
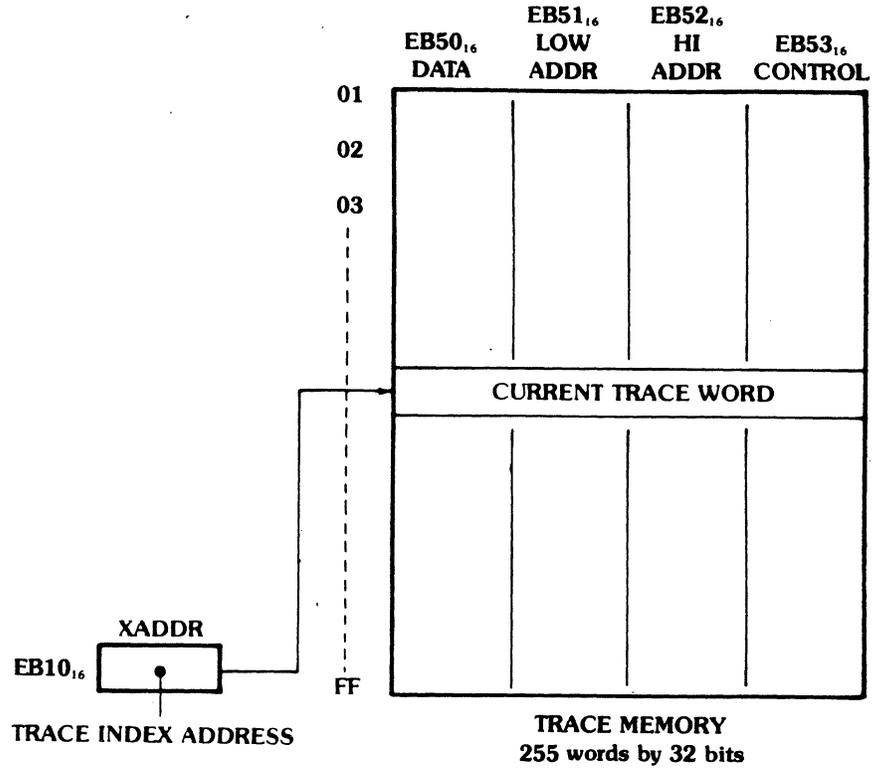
**TADDH (EB52<sub>1,6</sub>)** This port gives access to the eight-bit portion of the current trace word that records the high order eight-bits of the address bus. The control program may read or write this location.

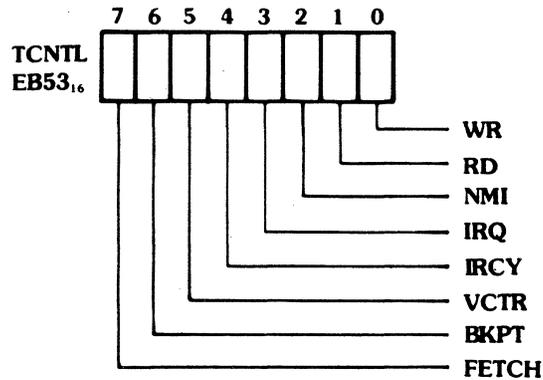
**TCNTL (EB53<sub>1,6</sub>)** This port gives access to the eight-bit portion of the current trace word that records the control bits of each machine cycle. The control program may read or write this location. The control bits are arranged in the port as shown:

# SECTION 8

## USER IMPLEMENTED CODE FUNCTIONS

Figure 8-2.5.





Any time that the control program writes new data into the Trace Memory, the data stored will immediately be seen on the appropriate displays.

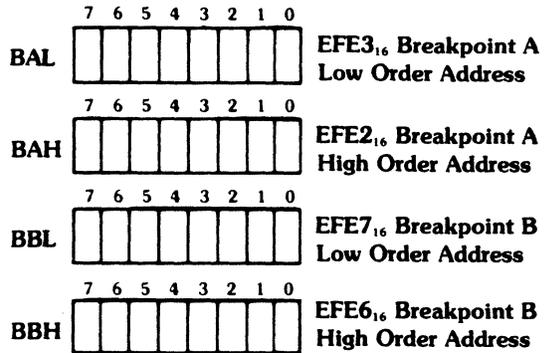
**SPEAKER:** The EM-186 incorporates a very small dynamic speaker located on the keyboard printed circuit board. Bit position 7 of the mode control register located at address  $EFE5_{16}$  is provided to control the current to the speaker to generate tones or other sounds under software control. It is necessary for the software to generate the actual waveform to be output by the speaker since there is no tone generation hardware in the EM-186. Writing a '0' to Bit 7 switches DC current to the speaker ON; writing '1' to Bit 7 switches the current OFF. The other bit positions of  $EFE5_{16}$  are detailed in the section on Breakpoint Comparators.

**HARDWARE CONTROLS:** There are a few additional output ports in the EM-186 internal address space that are used for such functions as initiating the Binary Address mode, entering the "introspection" mode and other hardware features. Details of these ports are not contained in this manual.

# SECTION 8

## USER IMPLEMENTED CODE FUNCTIONS

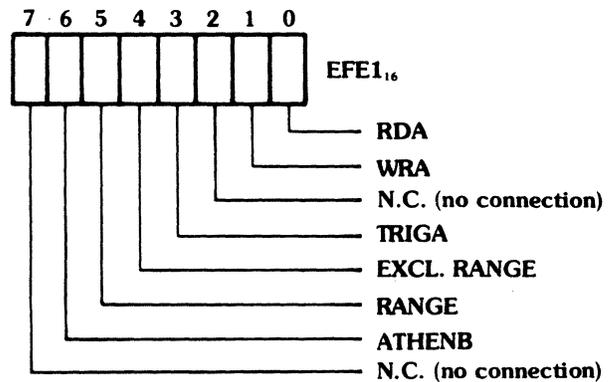
**BREAKPOINT COMPARATORS:** A series of registers is used to set up the Breakpoint Comparator address values and to control the desired operating mode. The address registers are shown below:

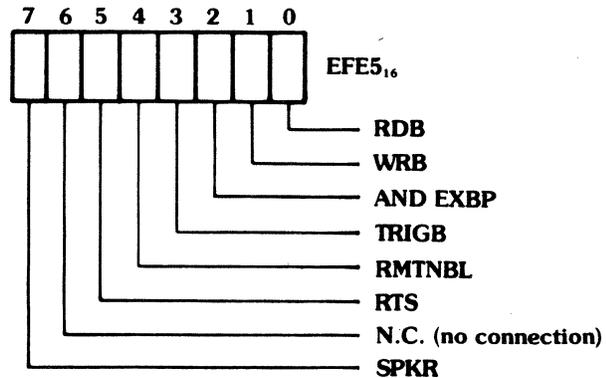


The registers BAL and BAH are used to set up the A Breakpoint address. The appropriate low and high order address bytes should be stored to these locations.

The registers BBL and BBH are used to set up the B Breakpoint address values in a similar manner.

The two registers that control the breakpoint mode are detailed below:





**RDA** and **RDB** are signals which enable the breakpoint comparators to respond to read cycles. Breakpoint comparators A and B will respond to read cycles if a '1' is stored to bit position 0 of **EFE1<sub>16</sub>** and **EFE5<sub>16</sub>** respectively.

**WRA** and **WRB** enable the breakpoint comparators to respond to write cycles by storing a '1' to bit position 1 of **EFE1<sub>16</sub>** and **EFE5<sub>16</sub>** respectively.

Notice that if '1' bits are stored to both the **RDA** and **WRA** ports, the A breakpoint comparator will respond to both read and write cycles. If a '0' is stored to both ports, the comparator will not respond to any cycles and is thus disabled. The B breakpoint comparator may be controlled in a similar manner.

**TRIGA** and **TRIGB** are used to output pulses to the BKPT A and BKPT B output pins of the Auxiliary Connector (pins 12 and 13; see Figure 9-1.1). Storing a '1' to the bit 3 position results in a high level at the corresponding output pin. To generate an output pulse, the software must store a '1' followed by a '0'. Inclusion of these signals in the EM-186 system enables user programmed Code Functions to generate trigger signals to external equipment such as oscilloscopes and signature analyzers.

**EXCL. RANGE**, **RANGE**, **ATHENB** are used to configure the breakpoint circuitry for several special operating modes (see Section 4-1.4). Any address outside the range A to B may be detected by writing a '1' to bit 4 of **EFE1<sub>16</sub>**. In order to include both end points of the range (address = A and address = B), it is necessary that both the A and B comparators be enabled for read and/or write cycles. Storing a '0' to **EXCL. RANGE**, any address in the range A to B will be detected. Again, it is required that the A and B comparators be set up to detect read and/or write cycles to ensure that both end points are included in the range.

# SECTION 8

## USER IMPLEMENTED CODE FUNCTIONS

**RANGE**, bit position 5 of  $EFE1_{16}$ , enables or disables the EXCL. RANGE circuitry by writing a '1' or a '0' respectively.

**ATHENB** sets up sequential operation of the comparators so that a breakpoint stop signal is generated when the B address is encountered after the A address has previously been encountered. To set up this mode, store a '1' to the bit 6 position of  $EFE1_{16}$ . Also, make sure that the RANGE circuitry is disabled ('0' to bit 5) and the A comparator is disabled ('0' to RDA and WRA bits) but the B comparator is enabled ('1' to RDB and WRB bits).

**AND EXBP**, bit 3 of  $EFE5_{16}$ , controls the external breakin function (pin 10 of the Auxiliary Connector). This bit is initialized to '0' upon powerup of the EM-186 allowing a low going input on pin 10 to stop program execution. Writing a '1' to bit 3 will disable this function.

**RMTNBL**, remote control enable, is not functional in the EM-186.

**RTS**, request to send, and **SPKR**, speaker, are not related to breakpoint control and are discussed elsewhere in this section.

NOTE: User programmed software can write to one or more of the bit positions of  $EFE1_{16}$  and  $EFE5_{16}$  registers without upsetting the values of the remaining bits by the technique of masking these bits. An example of such an operation is shown below:

### **:Disable 'A' breakpoint comparator**

```
LDAA $EFE1 : Load register image
ANDA #$FC  : Write '0' to RDA and WRA,
             mask other bits
STAA $EFE1 : Rewrite to RAM
```

### **8-3 ENTRY TO USER CODE FUNCTIONS**

The Code Functions that are built in to the EM-186 are all called with keystroke sequences that begin with one of the letter keys, such as CODE A1, CODE C4 and CODE D2. The Code Functions that use the decimal digit keys (0-9) are reserved for calling user programmed Code Functions. The keystroke sequences that are used to transfer control to user Code Functions are as follows:



Key Sequence

Transfer Address

CODE 0	D000 <sub>16</sub>
CODE 1	D003 <sub>16</sub>
CODE 2	D006 <sub>16</sub>
CODE 3	D009 <sub>16</sub>
CODE 4	D00C <sub>16</sub>
CODE 5	D00F <sub>16</sub>
CODE 6	D012 <sub>16</sub>
CODE 7	D015 <sub>16</sub>
CODE 8	D018 <sub>16</sub>
CODE 9	D01B <sub>16</sub>

Thus, each of the key sequences has associated with it an entry address in the address space assigned to the Diagnostic Prom socket. It is the responsibility of the user to properly code the EPROM so that the desired actions occur for each entry address. It is necessary that the first instruction of every Code Function be a jump instruction (op-code 7E<sub>16</sub>) because the control software examines the Diagnostic Prom for the presence of this data before transferring control to it. See the examples given in Section 8-8.

8-4 INTROSPECTION MODE

The EM-186 Diagnostic Emulator has been designed with a special feature that is primarily intended as an aid to testing and debugging Code Function programs that have been programmed into EPROMs and plugged into the Diagnostic Prom socket. This special feature is the "Introspection Mode" in which the EM-186 is caused to turn its attention to its own internal address space. In this way, the user may examine and store to the internal address space and, with certain limitations, may single step programs that execute in the internal address space.

### 8-4.1 CODE F

The introspection mode is entered by the key sequence:



After entering the CODE F mode, the user may examine or alter the internal memory space, step or run programs in the internal memory space, and review the contents of the trace memory after program execution. Breakpoints may also be used to halt program execution at appropriate internal addresses. The RESET key returns the EM-186 to normal operation.

### 8-5 GETTING TO AND FROM THE TARGET SYSTEM

The EM-186 Control Program, together with the built-in diagnostic routines and any userprogrammed code functions, executes within the EM-186 internal "protected" address space. As a consequence, programs in this internal address space do not have direct access to the target address space, but must make use of special hardware in the EM-186 logic to make the target address space accessible. Code Function Programs may have a requirement from time to time to do one of the following things:

1. Read from or write to a location in the target address space.
2. Go to and begin executing a program residing in the target address space.
3. Return from running a program in the target address space to a program (user code function routine) in the internal address space.

The following sections give detailed information on these functions.

#### 8-5.1 EXAMINE AND STORE

The simplest method of reading and writing data to the target system is to use subroutines that exist in the EM-186 control program. Two subroutines are provided, as follows:

**STMSR** Store the data contained in accumulator A to the target address specified in the index register.

**EXMSR** Load accumulator A from the target address specified in the index register.

The entry addresses of these subroutines (and other useful subroutines) are given in Section 8-6, User Accessible Subroutines.

The two subroutines shown above operate by performing a read or a write operation to the specified address after commanding the EM-186 hardware to make the target address space accessible during the transfer interval.

### 8-5.2 PAUSE to RUN

The EM-186 Control Program (firmware) is normally in control of the emulator when in the PAUSE mode. Depressing the RUN or RUN BKPT Key causes the control program to execute a sequence of operations that will load the processor registers with the values that had previously been saved (when the EM-186 last entered the PAUSE mode) and then does a coordinated jump to the target system program. The EM-186 hardware will switch to the target address space at the correct time for execution of the first instruction. It is possible for a userwritten Code Function program to jump to a target system program in the same manner. This facility allows a user Code Function to place programs into the target address space (either user RAM or Overlay RAM) and then transfer control to that program. The target system program will then proceed to execute from within the target address space in a normal manner.

The best way for the user to transfer control to a target system program is to use the existing EM-186 internal routine. The following steps are suggested:

1. Set up the user register save locations (addresses  $EFE8_{16}$  to  $FFFF_{16}$  in the scratchpad RAM) as desired. In particular, be sure to set the target program counter (RGPC) location to the correct starting address. Other registers may be used, if desired, as a means of passing parameters to the target program.
2. If the target program needs parameters or data from the internal program, then transfer the data to the target RAM using the STORE subroutine as appropriate.
3. Perform the transfer of control to the target program by jumping to the RUN routine at address  $FE28_{16}$ .

The RUN routine will load the processor registers, do the required coordination of the EM-186 hardware, and start the target program running with the desired register values initialized.

### 8-5.3 RUN to PAUSE

When a program is executing in the target address space and it is desired to transfer control into the internal software, there are only three ways available to cause this to occur. They are:

1. Reset the system.
2. Press the STEP key.
3. Cause a breakpoint to occur, either with one of the breakpoint comparators or by means of the external breakpoint input connection.

The first two methods are commonly used during manual operation; the third method may be used during manual operation or a sort of automatic operation in which it is desired that a Code Function set up the conditions to enable a target program to get back to the internal environment when it so desires.

### 8-5.4 REENTRY JUMP

Some applications require that the EM-186 control software transfer control to a user program each time emulation of the target program is halted. Such a program might be a "soft shutdown" program that prevents damage to the target system when execution is halted. (See Section 9-8, Soft Shutdown.) The EM-186 has the flexibility required to give control to a user written subroutine each time the RUN to PAUSE sequence of the emulator is executed. Normally, this subroutine would be programmed into an EPROM and inserted into the front panel socket of the EM-186.

In normal operation, the EM-186 executes an internal RUN to PAUSE routine each time the target program is halted. This routine first saves the processor registers in the scratchpad RAM save area, sets up the display to show the correct data, and finally goes to the keyboard input routine to determine the next action required. Before going to the keyboard routine, however, the RUN to PAUSE routine examines location  $EF9F_{16}$  to see if it contains a jump instruction opcode ( $7E_{16}$ ). If it does, the EM-186 will regard the jump instruction as the first instruction of a usersupplied subroutine, and will call the subroutine. (The EM-186 calls the address of the jump instruction which then jumps to the main body of the subroutine.)

The user-supplied subroutine will usually be located in the front panel EPROM, but may also be located in the user portion of the internal scratchpad RAM as the following example illustrates.

The following small program may be entered from the keyboard of the EM-186. It causes the EM-186 to beep each time it transfers from RUN to PAUSE. Enter the program with the following steps:

1. Reset the EM-186, then execute CODE F to place the emulator in "introspection" mode.
2. Enter the jump instruction:
  - at  $EF9F_{16}$  enter  $7E_{16}$ ;
  - at  $EFA0_{16}$  enter  $EC_{16}$ ;
  - at  $EFA1_{16}$  enter  $00_{16}$ .

These three bytes constitute a jump instruction to location  $EC00_{16}$  in the internal address space. Location  $EC00_{16}$  is the first location of the user portion of the scratchpad RAM.

3. At memory address  $EC00_{16}$ , enter the following four-byte program:
  - at  $EC00_{16}$  enter  $BD_{16}$ ;
  - at  $EC01_{16}$  enter  $FE_{16}$ ;
  - at  $EC02_{16}$  enter  $34_{16}$ ;
  - at  $EC03_{16}$  enter  $39_{16}$ .
4. Reset the emulator to exit the "introspection" mode and proceed to operate the emulator. Note that each time the emulator transfers from RUN to PAUSE, the beeper will sound.



In most practical cases, the user subroutine will be located in the front panel EPROM instead of RAM as was done in this example. Also, the jump instruction may be easily written into addresses EF9F through EFA1 by a Code Function program also residing in the EPROM. Executing the Code Function will enable the user subroutine and a second Code Function could be written to disable the subroutine by changing the jump instruction op-code to 00<sub>16</sub> (or any other code except 7E<sub>16</sub>).

**8-6 USER ACCESSIBLE  
SUBROUTINES**

The EM-186 Control Program contains handlers and subroutines which may be used by the user in constructing his own Code Functions. The entry addresses and functions of the routines are summarized in the following table.

Table 8-1. User Accessible Subroutines

ADDRESS	NAME	DESCRIPTION
FDE0 <sub>16</sub>	CODEFN	Execute the built-in Code Function designated by the contents of accumulator A. For example, if accumulator A is loaded with A8 <sub>16</sub> and this subroutine called, then the EM-186 will execute Code Function A8; if the Code Function completes successfully, this subroutine will return to the calling program.
FDE3 <sub>16</sub>	EXMSR	Load accumulator A from the target address specified in register X. (Condition codes and accumulator altered.)
FDE6 <sub>16</sub>	STMSR	Store the data contained in accumulator A to the target address specified in register X. (Condition codes are altered.)
FDf2 <sub>16</sub>	LWLMIT	Compares X to the BEGIN address; the subroutine returns with CY = 1 if X = BEG. (Conditions altered.)
FDf5 <sub>16</sub>	HILMIT	Compares X to the END address; the subroutine returns with CY = 1 if X = END. (Condition codes altered.)
FDf8 <sub>16</sub>	SIA	ASCII Serial Input. Serial data entered at the serial port is returned in accumulator A. Bit seven is always zero. Register A may be altered. When this routine is called, the RTS line (request-to-send) automatically goes high. The RTS line goes low again whenever the XECUTE routine is entered.
FDfB <sub>16</sub>	SIB	Binary Serial Input. Serial data entered at the serial port is returned in accumulator A. All eight bits are returned to the user without alteration. When this routine is called, the RTS line (request-to-send) automatically goes high. The RTS line goes low again whenever the XECUTE routine is entered.

# SECTION 8

## USER IMPLEMENTED CODE FUNCTIONS

ADDRESS	NAME	DESCRIPTION
FDFE <sub>16</sub>	RHB	<p>Read-Hex-Byte. This subroutine obtains two ASCII characters from the serial port. It checks to see if the characters represent valid hexadecimal digits. If they are, they are then converted to an eight-bit binary number and the subroutine returns to the calling program with this result in accumulator A. The value in accumulator A is also added to location "SUMCHK" (EFB4<sub>16</sub>).</p> <p>This routine calls "CNVHEX" (see "CNVHEX") for results of nonhex characters. This subroutine alters the accumulator and condition codes.</p>
FE01 <sub>16</sub>	CNVHEX	<p>Convert ASCII character to binary value. This subroutine expects an ASCII character in accumulator A that represents one of the digits 0, 1, 2, ..., 9 or one of the letters A, B, C, D, E, F. The character is converted into a numeric value corresponding to the ASCII-hex character in accumulator A. The routine returns with the value in accumulator A. If the ASCII character originally in accumulator A does not represent one of the hexadecimal digits, location "NON HEX" EFB6<sub>16</sub> is incremented as an error indicated, and the A accumulator is cleared.</p>
FE08 <sub>16</sub>	CRLF	<p>Outputs a line ending sequence to the RS-232C port which consists of a carriage return (0D<sub>16</sub>) line feed (0A<sub>16</sub>), and two null characters (00<sub>16</sub>). (Condition codes altered.)</p>
FE0D <sub>16</sub>	SO	<p>Data in accumulator A is sent to the serial port. In addition, this subroutine affects a one byte counting location that is reset to zero each time a carriage return code is output (0D<sub>16</sub>) and is incremented by one each time a printable ASCII character is output. This counting location is also affected and examined by the TAB routine as well as other output operations. This subroutine may also be called with a software interrupt followed by 6A<sub>16</sub>.</p>
FE13 <sub>16</sub>	BYTE	<p>Convert the contents of accumulator A to two ASCII characters representing the value in hexadecimal, then outputs these characters to the RS-232 port. No registers are altered. (Condition codes altered.)</p>
FE16 <sub>16</sub>	NIB	<p>This subroutine converts the low-order 4 bits of accumulator A to one ASCII character representing the value in hexadecimal and outputs the character to the serial port. (Condition codes altered.)</p>
FE19 <sub>16</sub>	NULL50	<p>Output 50 ASCII null characters (00<sub>16</sub>) to the serial interface. This routine is primarily useful to produce blank tape leader on a paper tape punch interfaced to the EM-186. (Accumulator A, B and condition codes altered.)</p>



ADDRESS	NAME	DESCRIPTION
FE1C <sub>16</sub>	ERR	Subroutine to report an error on the display. This routine shows the characters "EC" on the address display (meaning Error Code), displays the contents of accumulator B in the data display, and emits three beeps. The routine then waits for the user to depress some mode selection key. This subroutine does not return to the calling program.
FE22 <sub>16</sub>	XECUTE	This routine returns program control to the standard Diagnostic Emulator firmware and readies it for Keyboard input. If the user has changed the display, it will return to its former state. This routine does not return to the calling program.
FE25 <sub>16</sub>	STATE	This routine places the Diagnostic Emulator in the TRACE VIEW mode and makes it ready to accept input from the Keyboard. This routine does not return to the calling program. (See XECUTE routine.)
FE28 <sub>16</sub>	RUN	This routine loads the processor registers with the values in the scratchpad RAM save area and then does a jump to the target system. The address contained in the PC save location will be the address where target system execution will begin.
FE2B <sub>16</sub>	UKBDSN	User's Keyboard scan routine. Data representing the keyswitch depressed is returned in accumulator A. In addition, if the keyswitch depressed is one of the hexadecimal numeric keys, the carry bit (CY) is set to one (true) when the subroutine returns. This routine ignores the RUN, RUN BKPT and STEP keyswitches. The table below shows the data returned in accumulator A for each keyswitch depression.

# SECTION 8

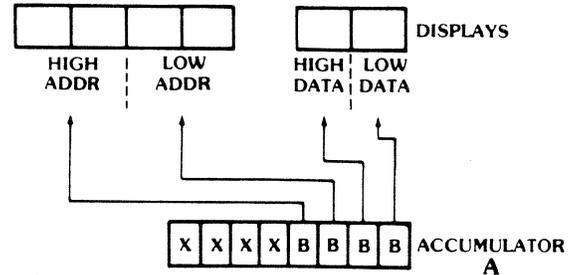
## USER IMPLEMENTED CODE FUNCTIONS

<u>KEYSWITCH</u>	<u>DATA</u>	<u>KEYSWITCH</u>	<u>DATA</u>
RESET	•	0	00 <sub>16</sub>
RUN		1	01 <sub>16</sub>
RUN BKPT	••	2	02 <sub>16</sub>
STEP		3	03 <sub>16</sub>
CODE	18 <sub>16</sub>	4	04 <sub>16</sub>
BKPT A	19 <sub>16</sub>	5	05 <sub>16</sub>
BKPT B	1A <sub>16</sub>	6	06 <sub>16</sub>
REG	1B <sub>16</sub>	7	07 <sub>16</sub>
MEM ADDR	14 <sub>16</sub>	8	08 <sub>16</sub>
BKBT TYPE	15 <sub>16</sub>	9	09 <sub>16</sub>
TRACE	16 <sub>16</sub>	A	0A <sub>16</sub>
LOAD DATA	17 <sub>16</sub>	B	0B <sub>16</sub>
STORE	10 <sub>16</sub>	C	0C <sub>16</sub>
EXAM	11 <sub>16</sub>	D	0D <sub>16</sub>
DEC	12 <sub>16</sub>	E	0E <sub>16</sub>
INC	13 <sub>16</sub>	F	0F <sub>16</sub>

• When RESET is depressed, the User's Code Function is aborted and the EM-186 reinitialized.

•• Keyswitches RUN, RUN BKPT and STEP are ignored by this routine.

ADDRESS	NAME	DESCRIPTION
FE3A <sub>16</sub>	DSPCTL	Display Control. This subroutine uses the low-order four-bits of accumulator A to control the blanking of the display digits. This is shown in the example below.



If a bit of accumulator A is a one when this subroutine is called, then the display digit or pair of digits corresponding to that bit illuminate(s). The accumulator and flags may be altered by this routine.

FE34 <sub>16</sub>	BEEP	A subroutine to beep the speaker located on the Keyboard printed circuit card. No registers are altered.
--------------------	------	--

In addition to the subroutines listed above, there are a number of Software Interrupt Subroutines in the EM-186 Firmware that may be accessed by the user. As with any SWI, the status of the MPU is saved in the stack and the interrupt mask bit is set. The entry addresses and functions of the SWI routines are summarized in Table 8-2.

Table 8-2. User Accessible Software Interrupt Subroutines.

ADDRESS	NAME	DESCRIPTION
3F60 <sub>16</sub>	PSHX	This SWI pushes the Index Register (IX) on the stack.
3F61 <sub>16</sub>	ADDAX	Add Accumulator A to Index Register.
3F62 <sub>16</sub>	ADDBAX	Add Accumulator B and A to the Index Register (B = high order byte, A = low order byte).
3F67 <sub>16</sub>	KBDSCN	Same as User Accessible Subroutine UKBDSCN (see Table 8-1).
3F68 <sub>16</sub>	EXGX	Exchange Index Register with top of stack.
3F69 <sub>16</sub>	TWEETS	This SWI causes the speaker to beep 3 times (used by the EM-186 control program as an error indicator).

# SECTION 8

## USER IMPLEMENTED CODE FUNCTIONS

### 8-7 INTERRUPTS

Target system interrupts are invisible to programs executing in the EM-186 internal environment. For this reason it is not possible to write Code Function programs that directly work with or test the user's interrupt system. Nevertheless, it is possible for a Code Function program to test or work with interrupts as follows:

1. The internal Code Function program, when it begins executing, first copies the interrupt portion of the routine to target system RAM. (If no RAM is available in the target system, the RAM Overlay may be used.)
2. The Code Function program sets up one of the breakpoint comparators to facilitate re-entry into the internal environment.
3. The Code Function program sets up the re-entry jump address in order to gain control after the breakpoint occurs.
4. The Code Function program transfers control to the program copied to the target system RAM.
5. When the breakpoint occurs, the internal program may read results left in RAM by the target system routine and take whatever additional action is desired. See Sections 8-5.3 and 8-5.4.

### 8-8 CODE FUNCTION EXAMPLE

An example of a Code Function program is given in this section. This example is a very simple routine that writes a range of target system memory to zeros.

#### EXAMPLE:

```
                STMSR EQU $F0E6
                ORGIN $D000
                JMP CODE0                :CODE 0 ENTRY

INITIALIZE
CODE 0          LDX, #6000                :INITIALIZE MEMORY POINTER
                LDAB, #00                 :INITIALIZE BYTE COUNT
:NOW LOOP TO CLEAR EACH TARGET MEMORY LOCATION
:FROM 6000H to 60FFH
CO              CLRA                      :CLEAR ACCUMULATOR A
                JSR STMSR                 JUMP TO SUBROUTINE
                INX                       INCREMENT INDEX REGISTER
                DECB                      DECREMENT ACCUMULATOR B
                BNE CO                    BRANCH IF NOT EQUAL TO ZERO
                RTS                       RETURN FROM SUBROUTINE
```

This example illustrates the following points:

1. The program originates at location D000<sub>16</sub> because this is the start of the address range allocated for the Diagnostic PROM.
2. The first instruction tells the program to jump to the actual starting point of the CODE 0 program. This jump instruction provides room for the other entry points, each having its own jump instruction. In this simple example, only one Code Function is implemented.
3. The Code Function program is written in standard 6800 assembly language.



4. Whenever the Code Function program wishes to access the target system memory space, it may most easily do so by using subroutines already present in the Diagnostic Emulator firmware. In this example, the STMSR subroutine is called to perform the write operation to the target system.
5. When the Code Function program has finished executing, it returns control to the Diagnostic Emulator firmware by executing a "RTS" instruction.
6. The EQU statements inform the assembler of the addresses of routines within the Diagnostic Emulator firmware—in this case, the addresses of the STMSR.

# SECTION 9

---

---

---

## SUPPLEMENTARY INFORMATION

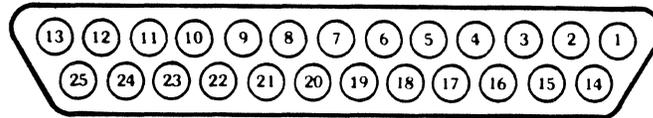
---

- 9-1 Auxiliary Connector**
  - 9-2 Option Switches**
  - 9-3 Serial Interface**
  - 9-4 Upload/Download Protocol**
  - 9-5 External Breakpoint**
  - 9-6 Trace Hold**
  - 9-7 Signature Analysis**
  - 9-8 Soft Shutdown**
-

## 9-1 AUXILIARY CONNECTOR

Figure 9-1.1. J3-Auxiliary  
Connector Pinout  
(D-Subminiature,  
Female)

The EM-186 Diagnostic Emulator has a back panel auxiliary connector that is used as the connection point for the RS-232C communications signals and various other signals. Figure 9-1.1 shows the pinout of the auxiliary connector and the signals present on each of the pins.



Pin		Pin	
1	PROTECTIVE GROUND	14	
2	SERIAL DATA (OUT) (RS-232C)	15	
3	SERIAL DATA (IN) (RS-232C)	16	
4	REQUEST-TO-SEND (OUT) (RS-232C)	17	
5	CLEAR-TO-SEND (IN) (RS-232C)	18	
6	DATA-SET-Ready (NOT IMPLEMENTED)	19	
7	SIGNAL GROUND (RS-232C)	20	DATA TERMINAL READY (OUT) (RS-232C)
8		21	
9		22	$\overline{\text{RUN}}$ (OUT)
10	EXT BREAK (IN)	23	
11	$\overline{\text{TRACE HOLD}}$ (IN)	24	
12	BKPT A and SA START (OUT)	25	SIGNATURE CLOCK (OUT)
13	BKPT B and SA STOP (OUT)		

**Pins shown without associated signals shown are not connected within the EM-186.**

The functions of the auxiliary connector signals are summarized below:

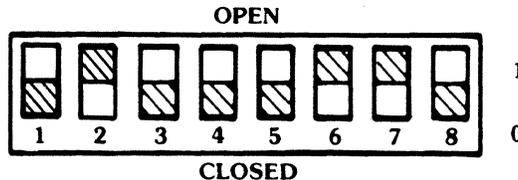
- Pin 1 Protective Ground:** Connected in the EM-186 to the chassis, and from the chassis to the protective ground terminal of the primary power input connector.
- Pin 2 Serial Data (Out):** This signal is driven to nominal  $\pm 12$  volt levels by an RS-232C compatible driver. See Section 9-3 (Serial Interface) for format of serial data.
- Pin 3 Serial Data (In):** The EM-186 accepts data on this pin that has voltage levels as specified by the EIA RS-232C specification and the format given in Section 9-3.
- Pin 4 Request to Send:** This signal is driven to nominal  $\pm 12$  volt levels by an RS-232C compatible driver. The state of this signal is determined by software in the EM-186.
- Pin 5 Clear to Send:** The EM-186 accepts a signal on this pin having RS-232C voltage levels. The state of this signal may be read by the EM-186 control software.

- Pin 7 Signal Ground:** Connected in the EM-186 to the system logic ground which is isolated from the protective ground (Pin 1). Note, however, that this ground is connected to the emulator probe ground pin; then when the EM-186 is connected to the target equipment, the target system logic ground and the EM-186 logic ground are connected together and to the ground system of the equipment plugged into the Auxiliary Connector.
- Pin 10 External Break (In):** A TTL level input with an internal 3.3K pull-up resistor. If this input is pulled low, the Diagnostic Emulator stops executing the target program as though STEP were depressed or an Internal Breakpoint were detected. (If the Diagnostic Emulator is already in PAUSE, this has no effect.) This input stops execution even when the breakpoints are not enabled.
- Pin 11 Trace Hold (In):** A TTL level input with an internal 3.3K pull-up resistor. If the Diagnostic Emulator is executing a target program and this input is pulled low, further updating of the Trace Memory stops, although the program continues to execute. The contents of the Trace Memory are effectively frozen, and can be reviewed later after program execution has been halted.
- Pin 12 BKPT A and SA START (Out):** A TTL level output providing a high-going pulse at the time breakpoint conditions are satisfied for the Breakpoint A Comparator. This signal can be used to trigger an oscilloscope at a particular point of program execution. It can also be used as the START signal for a signature analyzer. This signal may be set high or low under software control when the Diagnostic Emulator is in PAUSE. This permits diagnostic routines to generate sync pulses or signature analyzer START signals under direct program control.
- Pin 13 BKPT B and SA STOP (Out):** A TTL level output associated with the Breakpoint B Comparator. It is functionally identical with the BKPT A signal described above.
- Pin 20 DATA TERMINAL READY (Out):** This signal is driven to a nominal  $\pm 12$  volts to indicate that the EM-186 is ready to send data. Its signal state does not change.
- Pin 22 RUN (Out):** A TTL level output that is active (low) if the EM-186 is executing the target program or accessing the target address space.
- Pin 25 SIGNATURE CLOCK (Out):** A buffered  $\bar{\Phi}_2$  clock signal. It is primarily used as a clock for signature analysis testing of equipment for which the EM-186 provides the stimulus.



9-2 OPTION SWITCHES

The EM-186 has a set of eight small switches that are accessible from the back panel of the machine. These switches are used to select optional operating characteristics of the EM-186 reset circuitry and communications interface.



**SWITCH:**

**1 Soft Vector Switch.** If CLOSED (down), the emulation processor will fetch interrupt vectors from Target System Memory in the conventional manner. If OPEN (up), an 8-byte section of EM-186 scratch RAM will overlay the Target System ROM. In the event of an interrupt during target program execution, the emulation processor will fetch the interrupt vector image stored in scratch RAM. Since the vectors now reside in RAM (initialized at powerup of the EM-186), the user can alter them via the Register Keyswitches on the front panel. Once the soft vectors have been altered, they can be reinitialized by executing Code Function CF (see Section 7-3).

NOTE: If the target system utilizes a priority interrupt controller (PIC), switch 1 should be set in the CLOSED position for proper interrupt vectoring.

**2 Trace Hold Input Latch.** If OPEN, Code Function E4 is enabled (allowing Trace Memory to freeze upon detection of a breakpoint, see Section 7-5). Also, if Pin 11 (Trace Hold) of the auxiliary connector is pulled low, the Trace Memory will stop and remain stopped irrespective of a further change on Pin 11. If switch 2 is CLOSED, Code Function E4 is disabled. If Pin 11 is pulled low, the Trace Memory will freeze but only for as long as the input stays low. If Pin 11 goes high, Trace Memory will resume.

**3 If CLOSED,** EM-186 ignores clear-to-send (CTS) signal and communications software will output data at any time on operator command. If OPEN, EM-186 will output data only if clear-to-send is in the ON (positive) state.

**4 If CLOSED,** target system RESET signal will reset the EM-186 in the same manner as the RESET Key. If OPEN, target system RESET signal will reset EM-186 emulation MPU but the operator station will not be reset. This makes it possible to emulate systems in which the MPU is made to restart at intervals as part of the normal operation of the system.

**5** If CLOSED, EM-186 RESET signal (from RESET Key or power-on-reset) is sent to target system RESET through the MPU reset pin. If OPEN, no reset of the target system is attempted.

**6,7,8** Set up characteristics of serial communications interface as shown in Table 9-1.

Table 9-1. Setup Characteristics for Serial Port.

SW6	SW7	SW8	DATA BITS PER CHARACTER	STOP BITS
0	0	0	5 **	1
0	0	1	5 **	1½
1	0	0	6 **	1
1	0	1	6 **	2
0	1	0	7	1
0	1	1	7	2
1	1	0	8	1
1	1	1	8	2

Normal Setup

\* CLOSED 0, OPEN 1

\*\* Standard EM-186 communications software requires at least 7 bits for operation.

### 9-3 SERIAL INTERFACE

The EM-186 Serial Interface is compatible with the RS-232C standard pin conventions and signaling levels. The signals and connections are given in Section 9-1 (Auxiliary Connector).

The format of a serial word is shown in Figure 9-3.1. When no data is being transmitted, the Serial Data Out pin will be at the -12 volt level (marking). When the EM-186 sends a character, there will always be a START bit, followed by 5, 6, 7 or 8 DATA bits, and 1, 1.5 or 2 STOP bits. The number of DATA bits and STOP bits are selected by the Option Switches on the back panel. See Section 9-2 (Option Switches).

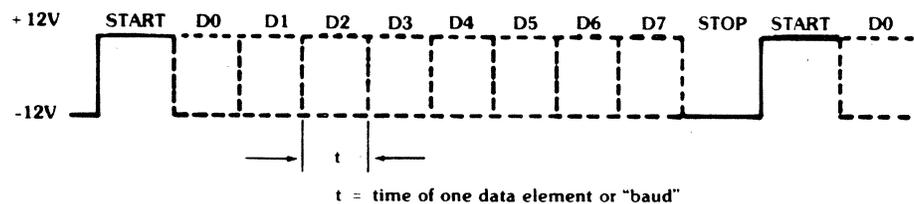
The standard EM-186 software transmits and receives ASCII characters which require 7 bits for their representation. For this reason, the option switches must be set for 7 or 8 bit characters for proper operation. Some data terminals require two stop bits for proper operation and the EM-186 will operate with these terminals; one stop bit is recommended for most other terminals because a somewhat higher data rate is obtained if time is not given to unneeded stop bits.

The EM-186 with standard software does not send or check parity. However, it is possible to have one of the data bits function as a parity bit if the parity generation and checking is done by software.

Two additional signals that are used by the EM-186 are the Request-to-Send (Pin 4) output and the Clear-to-Send (Pin 5) input. The EM-186 standard software uses these signals to coordinate the data transfer. When the EM-186 is ready to begin receiving data, it changes the Request-to-Send line from low to high and awaits data transmission. When the EM-186 has finished receiving data, it will return the Request-to-Send line to the low state. When the EM-186 is ready to send a character, the software tests the condition of the Clear-to-Send line and transmission of the character proceeds only if Clear-to-Send is in the high state; the character is held if the signal is in the low state. Thus, a receiving device may control the transfer of data by taking the Clear-to-Send line high when more data is desired and low when not ready for data. The EM-186 may be made to consider the Clear-to-Send line as always high by closing Option Switch 3 on the back panel.

The serial port transmission rate is controlled by the rotary hexadecimal switch in the lower left corner of the back panel. The EM-186 is capable of communicating at baud rates from 50 Baud to 19,200 Baud. See Figure 9-3.1.

Figure 9-3.1. Serial Word Format.

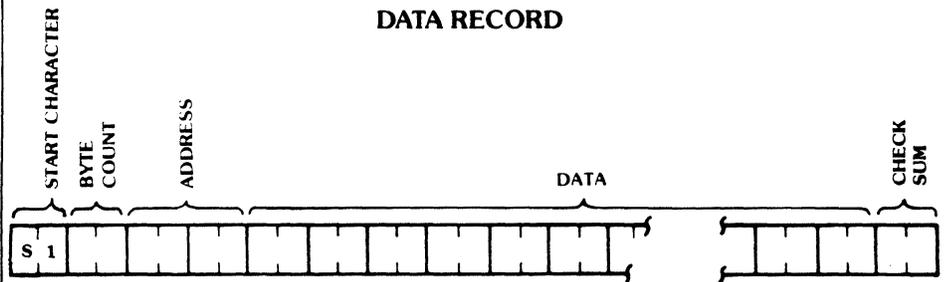


t = time of one data element or "baud"

Switch Position	Baud Rate	t	
D	50	20	mSEC
C	75	13.33	
0	110	9.09	
B	134.5	7.43	
1	150		
	56.67		
A	200	5	
2	300	3.33	
9	600	1.67	
4	1,200	833	uSEC
5	1,800	556	
3, 8	2,400	417	
6	4,800	208	
7	9,600	104	
E, F	19,200	52	

### 9-4 UPLOAD/DOWNLOAD

The EM-186 routines CODE C3 and CODE C4 initiate routines to load the target memory space with data from the serial link or dump data from the target address space to the serial link. The EM-186 uses a particular format to transfer the data. This format is compatible with the Motorola family of development systems.



#### START CHARACTER

An "S1" is used to signal the start of a record.

#### BYTE COUNT

The number of data bytes plus 3 (1 for checksum and 2 for address) in hexadecimal notation.

#### ADDRESS

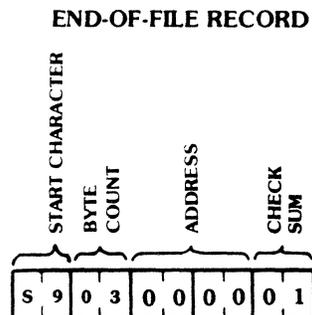
Four ASCII characters representing hexadecimal digits giving the address in target memory where the first of the data bytes of this record is to be located. The following bytes in the record are located in sequentially higher addresses in memory.

#### DATA

Each two ASCII characters representing hexadecimal digits give the bit pattern of one eight-bit byte of data. The total number of data bytes in the record is given by the byte count.

#### CHECK SUM

One's complement of the binary summation of the preceding bytes in the record (including byte count, address, and data bytes in hexadecimal notation).





**START CHARACTER**

An S9 is used to signal the start of a record.

**BYTE COUNT**

Byte Count = 03

**ADDRESS FIELD**

Four ASCII characters representing hexadecimal zeros, or the starting address of the program.

**CHECK SUM**

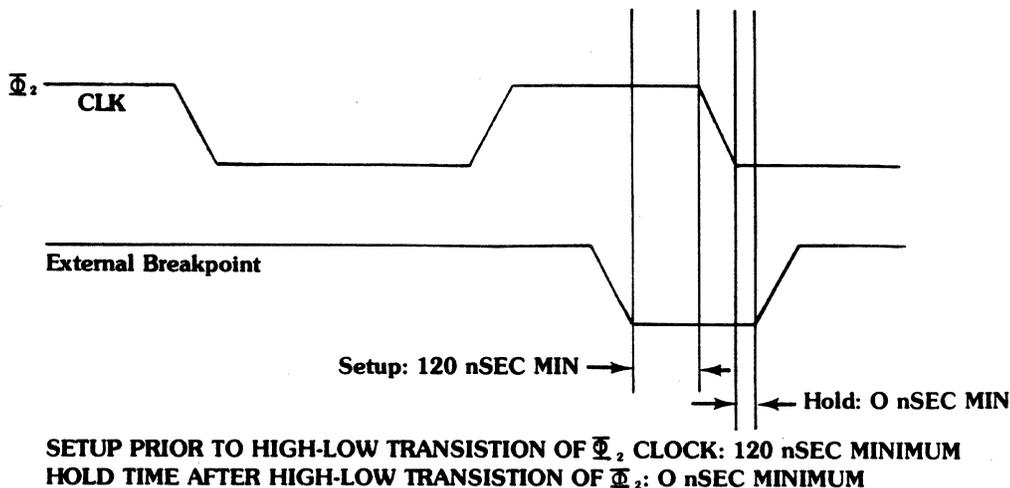
One's complement of the binary summation of the preceding bytes in record.

**9-5 EXTERNAL BREAKPOINT**

The EM-186 Diagnostic Emulator is provided with an input that permits an external signal to halt the execution of the target program when the EM-186 is in the RUN mode. Pin 10 of the back panel Auxiliary Connector (J3) is the input connection. External Breakpoint is a TTL level input with a 3.3K resistor pull up to +5 volts. If this input is in the high state, or if the input is left open, then the EM-186 will run the target program in the normal manner. If this input is pulled low, the target program will halt; if the target program is already halted, the External Breakpoint signal will have no effect.

The EM-186 samples the External Breakpoint input at the trailing edge (high-to-low transition) of the  $\Phi_2$  signal of the MPU. If the signal is low at the sample time, the signal is entered into the Trace Memory, thus marking the cycle during which the signal was detected; circuitry in the EM-186 is also armed to halt program execution after completion of the current instruction. When the target program has been halted, the EM-186 firmware will determine which cycle of the last instruction caused the breakpoint and the Trace Memory will be positioned to display that cycle. Figure 9-5.1 shows the timing relationships of the External Breakpoint signal.

Figure 9-5.1. Timing Relationships.



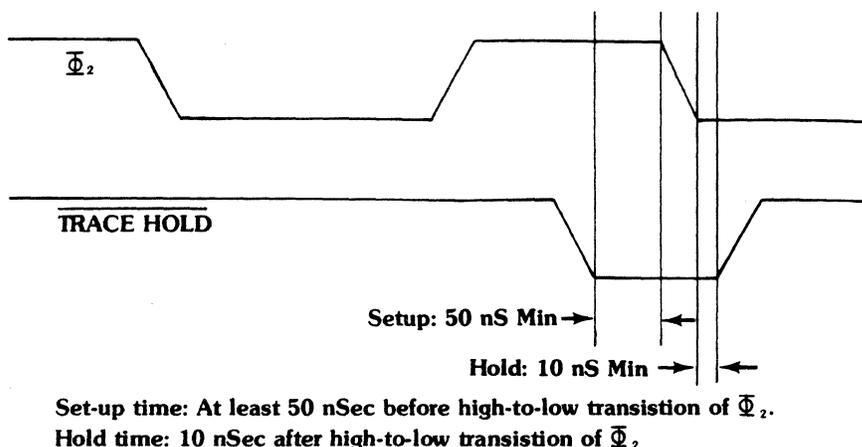
### 9-6 TRACE HOLD

The EM-186 Diagnostic Emulator is provided with an input that permits external equipment to control the tracing of program execution. Pin 11 of the back panel Auxiliary Connector (J3) is the Trace Hold input. Trace Hold is a TTL level input with a 3.3K resistor pull up to +5 Volts. If this input is in the high state, or if the input is left open, then the Trace Memory operates normally. If this input is pulled low, the Trace Memory stops tracing program execution.

The circuitry controlling the Trace Hold input must ensure that setup and hold time requirements are met for reliable operation. The requirements are shown in Figure 9-6.1.

The Trace Hold feature may be used to capture trace data on a selective basis as detailed in the following paragraphs.

Figure 9-6.1. Trace Hold and Timing



#### 9-6.1 WINDOW MODE

Window Mode operation can be implemented by an external circuit that holds Pin 11 low except for the duration of the desired trace period. For this mode it is necessary that switch 2 on the back panel be in the CLOSED position (down).

#### 9-6.2 HOLD TRACE ON BREAKPOINT

Sometimes it is desirable to freeze the Trace Memory at a particular point in the program while continuing program execution. To implement this feature, execute Code Function E4 and enter the desired Breakpoint configuration on the front panel. Press RUN to start program execution. Once the Breakpoint has been detected, the Trace Memory will stop tracing and the display will freeze showing the Breakpoint Address.

NOTE: Switch 2 on the back panel must be in the OPEN (up) position.

#### 9-6.3 SELECTIVE TRACE

External circuitry may be designed for a variety of selective tracing functions. An example is an application in which it is desired to use the capacity of the Trace Memory to capture cycles written to a particular I/O port; the I/O port select signal (port decode) may be routed to the Trace Hold input to permit Trace Memory operation only when the port select signal is active. When execution is halted and Trace Memory contents are reviewed, only bus cycles to the I/O port will be seen.



## 9-7 SIGNATURE ANALYSIS

In 1977, the Hewlett Packard Company introduced a digital servicing technique called signature analysis. The signature analysis technique requires first of all that the system under test be stimulated to cause repetitive patterns or bit streams to occur at various circuit nodes of the system. When such a stimulus is applied, it is possible to use an instrument such as the HP 5004A Signature Analyzer to observe these bit streams and convert them to four-digit hexadecimal displays on the front panel of the instrument. The bit stream, or pattern of lows and highs present at a given point in a circuit, is the "signature" of that circuit node. Faulty components, opens, shorts and other circuit defects will almost always cause alteration of a signature that may be observed by the signature analyzer.

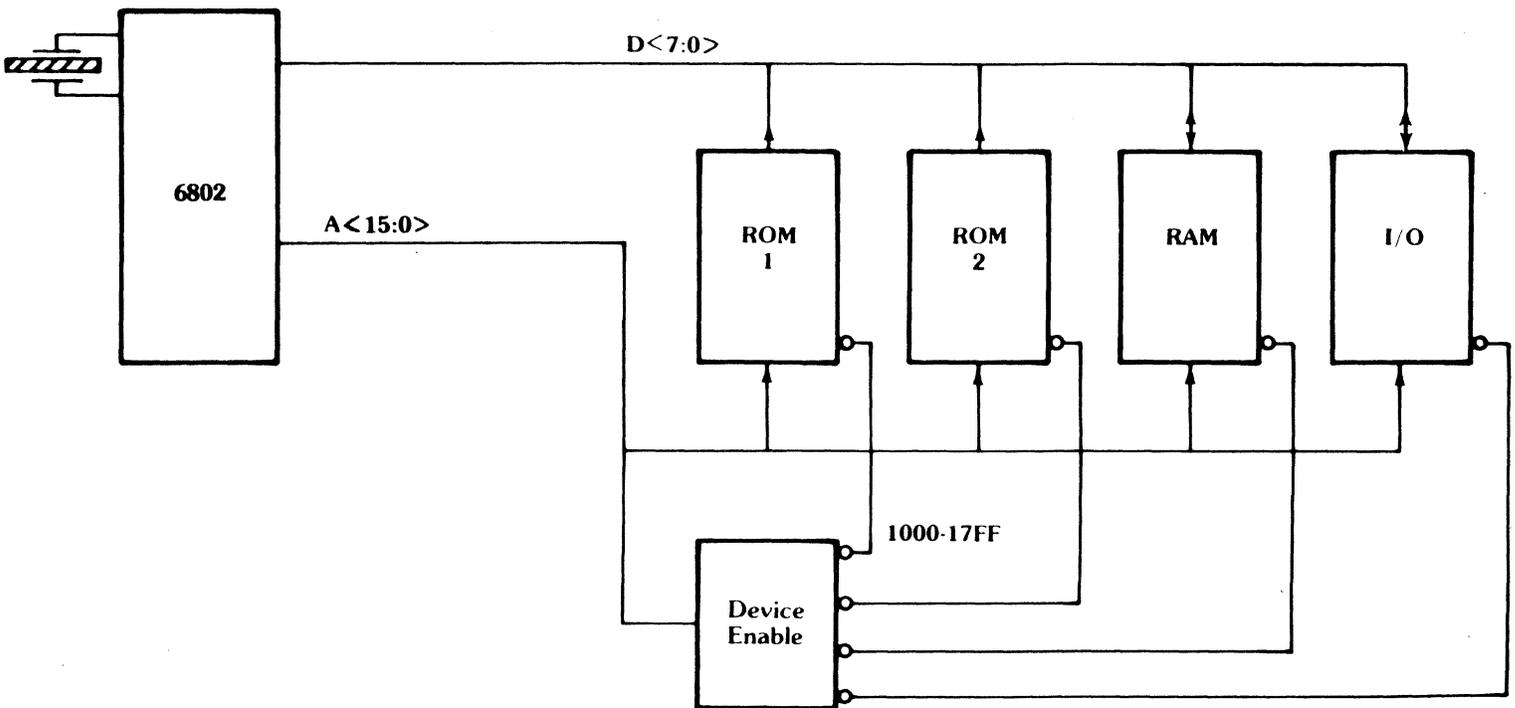
The EM-186 Diagnostic Emulator does not contain circuitry for examining signatures at circuit nodes. It does, however, contain preprogrammed stimulus routines that may be used to generate the repetitive signals that must be present for the signature analysis concept to work.

Figure 9.7-1 is a simplified microprocessor system diagram and shows an 6802 processor two ROMs, one RAM, some I/O circuitry and device enable logic. To test a system such as this one, first perform the obvious checks such as measurement of the supply voltages and then connect the EM-186 to the circuit. The system clock may be checked by using the CODE D2 function; the clock frequency displayed by the EM-186 should be onehalf of the crystal frequency. Now proceed with signature analysis testing by connecting a signature analyzer (such as the HP 5004A) to the EM-186 Auxiliary Connector (J3, on the back panel of the EM-186) as follows:

SA GROUND	to J3 - 24	(GROUND)
SA START	to J3 - 12	(BKPT A and SA START)
SA STOP	to J3 - 13	(BKPT B and SA STOP)
SA CLOCK	to J3 - 25	(SIGNATURE CLOCK)

The selector switches for the START, STOP and CLOCK signals on the signature analyzer should be set for low-to-high edge recognition (buttons out on the HP 5004A).

Figure 9.7.1 Simplified  
microprocessor system  
diagram.





### BINARY ADDRESS TEST (FREE-RUN)

The EM-186 Diagnostic Emulator has a built-in mode which places the MPU in a free-run condition by inserting continuous NOP instructions into the MPU. As a result, the processor outputs successive addresses, along with the RD signal; this mode is useful as a signature analysis stimulus routine because it stimulates all of the address lines in the system and since the RD signal is active, any devices in an operating system will drive the data bus when appropriate addresses are present.

To start with, set both the A and B breakpoint comparators so that they will respond to READ cycles at address  $0000_{16}$  (See Section 4-1.4). Next, start the binary address routine by depressing the keys for CODE B5. The EM-186 will begin to output incrementing addresses; a SA START pulse and an SA STOP pulse will occur each time address  $0000_{16}$  is output. The stimulus and signature analyzer are now ready for use.

At this point, the address bus signals may be probed with the signature analyzer and each should display its characteristic signature. The various device enable signals may be probed and, if the system circuitry is working correctly, characteristic signatures will be obtained. Various nodes internal to the Device Enable Logic may also be probed; in short, any circuit point may be tested where the signal present is determined by the address inputs and the RD signal.

In most cases, the data bus cannot be tested with this setup because the data bus signals are not determined by addresses for all possible address values. For example, some addresses may result in floating the data bus; other addresses may select RAMs whose contents are not known. Therefore, to test the data bus using the signature analysis technique, it is necessary to restrict the start-stop window of the signature analyzer so that the data bus is sampled only when addresses are present that should result in known data on the bus.

Suppose that it is known that ROM 1, in Figure 9.7-1, is enabled by the Device Enable Logic for any address in the range from  $1000_{16}$  to  $17FF_{16}$ . If the SA START signal could be generated when the incrementing address reaches  $1000_{16}$  and the SA STOP signal generated when the incrementing address reaches  $17FF_{16}$ , then signatures would be computed only during the time the data bus contained deterministic data. The SA START and SA STOP signals may be easily adjusted to occur at any desired addresses by setting the appropriate breakpoint values into the A and B breakpoint comparators. For the example just given, set the A comparator to respond to READ cycles at address  $1000_{16}$  and set the B comparator to respond to READ cycles at address  $17FF_{16}$ . Then test the eight data lines to obtain the characteristic signatures. Note that the signatures obtained depend not only on the details of the circuitry of the system under test, but also on the contents of the ROM involved; consequently, this test also verifies that the ROM contains the same pattern as the ROM for which the reference signatures were originally obtained.

The EM-186 also has a built-in test function for obtaining a signature of a ROM in a system, and no signature analyzer is needed. The test is set up by entering the first and last address of the ROM into the BEG and END registers of the EM-186 to define the range over which the routine will operate. Then start the routine with the Keys for CODE D3. The routine will execute and then display a four-digit hexadecimal signature on the EM-186 front panel. The signature obtained does not have any simple relationship to signatures obtained with the HP 5004A; for one thing, the CODE D3 algorithm operates on all eight data bits of the ROM word simultaneously while the eight signatures obtained by the HP 5004A for a ROM are computed from one "bit slice" of the ROM at a time. In addition, the generating polynomial used by the EM-186 routine differs from that used by the HP 5004A. See Section 7-4 for additional information.

Other routines that are programmed in the EM-186 Diagnostic Emulator may be useful as stimulus routines for signature analysis testing. The CODE B4 routine repetitively stores a data pattern and the complement of that pattern to a selected address.

In special cases it may be found necessary to write custom CODE function routines to stimulate a system in a way that useful signatures may be obtained. As an example, consider the problem of obtaining a signature at the outputs of an LSI interface chip such as the Motorola PIO. This device requires that various control registers and data direction bits be set up for the intended application before data transfers are performed. A custom CODE Function routine can easily perform the desired set-up and then generate the stimulus for signature analyzer probing.

For additional information on Signature Analysis testing, see the following publications:

1. "Hexadecimal Signatures Identify Troublespots in Microprocessor Systems", Gary Gordon and Hans Nadig. ELECTRONICS, March 3, 1977.
2. Application Note 222, "A Designer's Guide to Signature Analysis", Hewlett Packard Corporation.

### 9-8 SOFT SHUTDOWN

In some applications it is desirable to halt emulation of the target program when a particular event occurs or when a particular address is reached; after this, it is necessary that the processor execute a program to shut down the target system equipment in an orderly manner. For example, there may be hammer driver coils which would burn up if left energized. The EM-186 may be configured to operate in systems with requirements like these by writing the needed soft shutdown routines and programming them into an EPROM that is then plugged into the Diagnostic PROM socket. A small Code Function program is also required to insert the re-entry jump instruction into the EM-186 internal scratchpad RAM (see Section 8-5.4). When enabled, the soft shutdown routine would be executed every time the emulator transfers from RUN to PAUSE, and also after each single-step instruction execution. After executing, the soft-shutdown subroutine should exit to the monitor routine with a return instruction.



# SECTION 10

---

---

---

## **MAINTENANCE & TROUBLESHOOTING**

### **10.1 MAINTENANCE**

**10.1.1 Power Supply**

**10.1.2 Cables**

**10.1.3 Probe Tip Assembly**

### **10.2 TROUBLESHOOTING**

### **10.3 PARTS LIST**

---

---

---

## 10.1 MAINTENANCE

Maintenance of the EM-Series Diagnostic Emulator has been minimized by the extensive use of solid-state components throughout the instrument. There are only three areas where you need concern yourself with maintenance.

- Power Supply
- Cables
- Probe Tip Assembly

These are discussed in the following paragraphs. In addition, a troubleshooting chart follows, with section references for this manual.

### 10.1.1 POWER SUPPLY

The power supply provides the necessary voltages to operate all of the logic contained within the Operator's Station, as well as the POD assembly. The power supply is a regulated supply that is adjustable via the potentiometer located on the regulator PC board within the operator station. The +5 volt supply can be checked by using a Digital Volt Meter (DVM) and applying the ground probe to pin 12 of the ZIF PROM socket and the positive probe to pin 24 of the ZIF PROM socket. The voltage can be adjusted with the potentiometer on the PC card.

To access the regulator PC card, remove the four exposed screws on the bottom on the Operator Station and then remove the top cover. The regulator PC card is located on the right-hand side of the instrument, with the potentiometer in the uppermost corner. Adjust the potentiometer to deliver  $5.00 \pm .05$  VDC. If you cannot adjust the potentiometer to this specification, contact the Applied Microsystems Corporation Technical Services Department.

### 10.1.2 CABLES

The interconnect cables are the most vulnerable area of the instrument due to constant flexing during insertion and extraction. First, inspect the cables for any obvious damage, such as cuts, breaks, or tears. Even if you have inspected the cables and cannot find any damage, there may be broken wires within the cables (usually located close to the ends). A broken wire within the cable will cause the instrument to run erratically or intermittently if the cables are flexed during the "RUN" mode. By swapping the cables in question with a known good set of cables, you can easily isolate the faulty cable. The parts list at the end of this section contains cable part numbers if you need to order replacements.

### 10.1.3 PROBE TIP ASSEMBLY

The Probe Tip Assembly is the small DIP header assembly that plugs into the target system CPU socket. The most obvious area to inspect is the 40-pin adapter as the pins can be broken during insertion or extraction. If one of the pins should be inadvertently broken, you should replace the complete 40-pin adapter.

#### NOTE:

The 40-pin adapter can be protected by installing a CPU socket (male-female) onto the 40-pin adapter. If a pin is then broken on the CPU socket, it is easier to replace because of its common usage.

You should also inspect the probe tip assembly to see if any of the 1/8 watt resistors have been broken.

**NOTE:**

Due to the close physical tolerance surrounding the 1/8 watt resistors, we recommend that they be returned to the factory for repair.

**10.2 TROUBLESHOOTING**

Troubleshooting microprocessor-based equipment can be a complex process, due mainly to the complex nature of several peripheral devices, such as the data and address lines. To assist you in identifying the faulty PC card or possibly a component, your emulator is equipped with diagnostic test routines. The diagnostic programs are described in Section 7; if you need to perform any specific test, you should refer to the description in Section 7. Before starting troubleshooting procedures, be sure that interconnect cables are installed properly in a compatible target system, with power applied to both the target system and the emulator.

The most common problems encountered are listed in Table 10-2. We recommend that you contact the Technical Services Department of Applied Microsystems Corporation if you experience any problems that do not fall within this range of items.

**NOTE:**

We do not recommend a component-level repair in the field, unless performed by a qualified service engineer.

**Table 10.2  
Troubleshooting**

<b>SYMPTOM</b>	<b>POSSIBLE CAUSES</b>	<b>SECTION</b>
Target system runs erratically	1. Faulty interconnect cables	10.1
	2. Intermittent contact on Probe Tip Assembly PC Card	*
	3. Broken pin on 40-pin adapter	3.3 10.1.3
	4. Power supply out of adjustment	1.4
	5. "Hold-tites" on Probe Tip Assembly missing (for connection to 40-pin adapter)	*
	6. Broken resistor on Probe Tip Assembly	10.1.3
	7. Option switches set improperly	9.2
	8. RAM Overlay switch on but memory not programmed	5.4

\*Call Applied Microsystems (Technical Services Department)

\*\*Check Target System

**Table 10.2**  
**Troubleshooting**

SYMPTOM	POSSIBLE CAUSES	SECTION
	9. Control PROMS on keyboard need to be reseated	7.4 (D-D)
	10. Emulator and target system not compatible	1.1
	11. Faulty address of data buffer (see Code Function A4 and B2)	7.1 (A-4) 7.2 (B-2)
Emulator will not communicate over RS-232 line	1. Option switches for character format set incorrectly	9.2
	2. Baud rate set incorrectly	9.3
	3. Target system requires a "null" modem cable (pin 2 and pin 3 of RS-232 connector reversed)	9.1 9.3
Target system will not run	1. Cables plugged in wrong	3.3
	2. Power supply out of adjustment	1.4
	3. Faulty interconnect cables	10.1
	4. Broken pin on 40-pin adapter	10.1.3 3.3
	5. Broken pin on interconnect cable connector	*
	6. RAM Overlay switch on, but memory not programmed	5.4
	7. No clock in target system	3.3
	8. No power (+5 volts) in target system	**
	9. Option switches set improperly	9.2
	10. Emulator and target system not compatible	1.1
	11. RUN key bad	*
	12. Constant target reset	3.3
	13. Target system has one or more DMA devices requesting the bus (Example: Bus Req line = True)	**

\*Call Applied Microsystems (Technical Services Department)

\*\*Check Target System

---

---

**10.3 PARTS LIST**

The following parts are available for you to order:

	Part Number
40-Pin Adapter	210-11410
Short Cable Set	600-11284
Long Cable Set	600-10653-01
Key Switch	510-10128
Hex Display	370-10009



---

# SET-UP CHECKLIST

Please read this checklist completely before using your new Applied Microsystems' Diagnostic Emulator.

You should review the following items:

1. **Are you using RAM overlay?** If so, check the manual for the following settings:
  - Enable switch (5.3)
  - Option switches (9.2)
  - RAM overlay memory address switches (5.3)
2. **How will you be using the RESET key?** See Section 9.2.
3. **Do you need to set the option switches?** If so, see Section 9.2
4. **Have you reviewed the specifications for the serial interface port?** This must be set properly to avoid damage, as there are extra, *non-RS232* signals on the connector. See Section 9.3.
5. **If using communications without a modem, you must use a null modem cable** (pins 2 and 3 are reversed). See Section 9.1.
6. **You may wish to protect the 40-pin adapter on the Probe Tip Assembly by installing a low-cost solderable CPU socket\* (male-female) onto the 40-pin adapter.** If a pin is then broken on the CPU socket, it is easier to replace because of its common usage.
7. **At a minimum, you should read sections applicable to the steps listed here, plus:**
  - **Section 1 – Introduction**
  - **Section 2 – System Components**
  - **Section 3 – Basic Operating Instructions**

If you experience difficulty in setting up your diagnostic Emulator, **review Section 3 and Section 10, Maintenance and Troubleshooting.** Section 3 tells you who to call for help if you have a problem setting up.

\* Ansley SMO-40-SGT, Samtec ICO-640-SGT, EMT 10640-01-446, Texas Instruments C72 4059, Robinson Nugent ICT-406-S-TG.

**PLACE CHECKLIST INSIDE FRONT COVER OF MANUAL  
FOR FUTURE REFERENCE**



**5020 148th Avenue N.E.  
P.O. Box C-1002  
Redmond, WA 98052  
(206) 882-2000  
Toll Free Service: 1-800-426-3925**

**920-10649-50  
January 1984**



# INDEX

## A

Auxiliary connector, 2-5, 9-2

## B

Back panel controls, 2-5  
Baud rate selector switch, 2-5  
Breakpoint system, 4-6  
Breakpoint comparators, 8-13

## C

Cables, 1-3, 2-3, 10-2  
Check byte, 9-8  
Code functions, 7-2  
Connection to target equipment, 3-2  
Communications, 9-2

## D

Data transmission, 9-2  
Default parameters, 7-16  
Diagnostic EPROM socket, 2-2  
Diagnostic functions, 7-2  
Disassembler, 2-6  
Disassembly format, 6-5  
Display panel, 2-2  
Displays, front panel, 2-3, 4-4  
Downloading  
  Protocols, 9-7  
  Protocol selection, 9-4  
  RAM overlay from front panel EPROM, 5-6  
  RAM overlay from serial link, 5-6  
  RAM overlay from target memory, 5-7

## E

Emulator probe, 2-2  
Entry to user code functions, 8-15  
EPROM socket, 8-5  
Error Codes, 7-9, B-1  
Examine and store, 8-17  
Examination and alteration  
  Memory locations, 4-11  
  MPU registers, 4-10  
  I/O ports, 4-15  
External breakpoint, 9-8  
Execution and control, 4-2

## F

Flags, 6-5  
Format definition, disassembly, 6-3  
Functions, EM-186, 4-2

## G

Getting started, 3-2

## H

Hexadecimal displays and trace memory, 8-9

## I

I/O devices, 8-5  
I/O ports, 4-15  
Installing RAM overlay, 5-3  
Interface chips, 9-5  
Introspection mode, 7-17, 8-16  
Internal environment of EM-186, 8-3  
Interrupts, 8-25

## J

Jump, re-entry, 8-19, 9-14

## K

Keyboard, 2-2, 8-5

## L

## M

Main power switch, 2-5  
Maintenance, 10-2  
Memory  
  Map, 8-3  
  Overlay, 8-2  
  Trace, 4-8  
Memory tests, 7-3  
MPU registers, 4-10

N

O

Operator's station, 2-2  
Option switches, 9-4  
Overlay memory, 5-7

P

PAUSE to RUN, 8-18  
Protocol selection, 9-4  
Protocols, 9-7

Q

R

RAM overlay, 2-5, 5-2  
RAM overlay: uploading/downloading, 5-6  
Read cycle control, 5-4  
Re-entry jump, 8-19  
Register examination, 4-10  
RESET keyswitch, 4-2  
Reset control, 9-4  
RUN keyswitch, 4-2  
RUN BKPT keyswitch, 4-3  
RUN to PAUSE, 8-18

S

Safety information, 3-2  
Scratchpad RAM, 8-5  
Scope loops, 7-6  
Selective trace, 9-9  
Serial connector, 2-5, 9-2  
Serial interface, 9-5  
Signature analysis, 9-10  
Speaker, 8-12  
Specifications, 1-3

Soft shutdown, 9-13  
Start character, 9-7, 9-8  
STEP keyswitch, 4-3  
Subroutines, user-accessible, 8-20

T

Trace hold, 9-9  
Trace memory, 2-2, 4-8, 8-11  
and hex displays, 8-9  
Trace window, 9-9  
Transparency, 1-2  
Troubleshooting, 10-3

U

Uploading  
Protocols, 9-7  
Protocol selection, 9-4  
RAM overlay from front panel EPROM, 5-6  
RAM overlay from serial link, 5-6  
RAM overlay from target memory, 5-7  
User-accessible subroutines, 8-20  
User PROM socket, 8-4

V

Voltage, operating, 3-2

W

Window mode, trace hold, 9-9

X

Y

Z