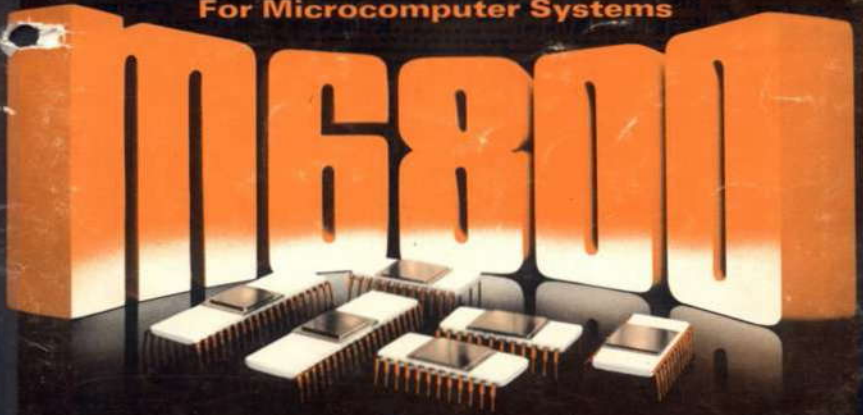
MOTOROLA

M6800 EXORciser User's Guide

Benchmark Family
For Microcomputer Systems



by Support Products Group

MOTOROLA
Semiconductor Products Inc.

M6800 EXORciser User's Guide

First Edition

Printed in Switzerland

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

LIST OF ILLUSTRATIONS

LIST OF TABLES

FIGURE 1-1. M6800 EXORciser (M68SDT)

CHAPTER 1 GENERAL DESCRIPTION

1-1 INTRODUCTION

The M6800 EXORciser (M68SDT) is a system development tool used in the design and development of M6800 Microcomputer Systems. The EXORciser Debug and the user's system are built around the M6800 Microcomputer Family of Parts. The M6800 Microcomputer Family of Parts are discussed in the following documents.

- M6800 Microprocessor Programming Manual
- M6800 Microprocessor Applications Manual
- M6800 Microprocessor Family of Parts Data Sheets

The EXORciser may be configured in a variety of applications and with various EXORciser options. This manual, rather than discussing every possible configuration and option, discusses only the basic EXORciser with each option except the Wirewrap and Extender Modules discussed in a supplement to this manual. The basic EXORciser is discussed in Paragraph 1.5 and the options are identified in Paragraph 1.6.

This manual provides general information, installation instructions, applications information, operating procedures, and theory of operation, for Motorola's M6800 EXORciser. The M6800 EXORciser is illustrated in Figure 1-1.

1-2 EXORciser FEATURES

The features of the basic EXORciser include:

- Flexible, adaptable, and expandable design development tool
- Easy to use
- Provides the Microprocessing Unit capability for both the EXORciser and the user's system
- Saves system design and development time
- Decreases system design and development costs
- Evaluates and debugs the user's program
- Evaluates and debugs the user's system hardware.

1-3 EXORciser SPECIFICATIONS

Table 1-1 identifies the basic EXORciser specifications.

TABLE 1-1. Basic EXORciser Specifications

CHARACTERISTICS	SPECIFICATIONS
Power Requirements	95-135/205-250 VAC 47-420 Hz, 250W
Word Size Data Address Instructions	 8 bits 16 bits 8, 16, or 24 bits
Memory Size	64k bytes
Instruction Set	72 variable length instructions
Clock cycle time	Selectable, 1μs or an external clock between 1 and 10μs
Interrupt	Maskable real time interrupt
Physicial Characteristics Table top Length Depth Height	 19.25 in. 17.50 in. 7.00 in.
Rack Mountable Length Depth Height	 19.00 in. 17.00 in. 7.00 in.
Baud Rates (Switch Selectable)	110, 150, 300, 600, 1200, 2400, 4800, and 9600

1-4 FUNCTIONAL DESCRIPTION

The M6800 EXORciser, illustrated in Figure 1-2, is an efficient and economical off-the-shelf systems development tool for the M6800 Microcomputer Family of Parts. The EXORciser may be easily tailored to meet the user's needs in the design and development of his system. Its modular design reduces the time required to develop a system and at the same time, provides great flexibility in configuring an emulation (functional representation) of the user's system. The EXORciser's EXbug Firmware, through its debug and program control features, minimizes the time required to develop user's systems. The EXbug firmware provides the EXORciser with the capability to:

- Display the contents of the MPU registers
- Step through the program
- Trace through user's programs to locate problem areas
- Stop the program on a selected program step
- Provide an oscilloscope trigger signal on a selected program step
- Abort from the user's program and return to the EXbug control program on command
- Reinitialize the EXORciser on command
- Change the contents of memory

The user communicates with the EXORciser in one of two ways.

- Through a RS-232C or TTY terminal
- Through the EXORciser front panel controls and indicators

The terminal device permits the user to communicate directly with the EXORciser's EXbug Firmware. The EXORciser's front panel permits the user to apply power to the EXORciser, to abort (exit) the EXORciser from a routine, and to initialize and restart the EXORciser. The EXORciser's unique front panel was designed to incorporate future EXORciser options such as a data keys and displays.

1-5 BASIC EXORciser SYSTEM (FIGURES 1-2 AND 1-3)

The basic EXORciser (M68SDT) consists of the MPU Module, the Debug Module, the Baud Rate Module, the Power Supply, and the chassis. These modules are built around the M6800 Microcomputer Family of Parts (MC6800 Microprocessing Unit, MC6820 Peripheral Interface Adapter, MC6850 Asynchronous Communications Interface Adapter, MCM6810 Random Access Memory, and MCM6830 Read Only Memory devices).

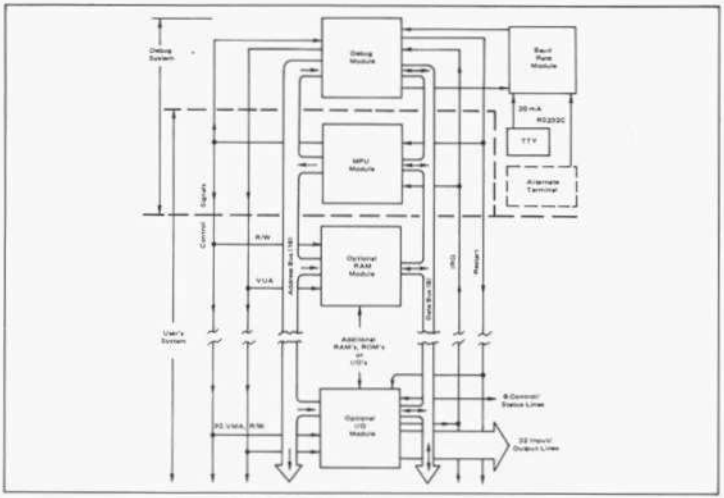


FIGURE 1-2. EXORciser Simplified Block Diagram

The MPU Module (Figure 1-4) incorporates the MC6800 Microcomputer Unit (MPU) and the system clock. This module, as illustrated in Figure 1-2, serves a dual function in that the module provides the MPU and clock for both the EXORciser Debug and the user's system. The MPU Module also initiates an EXORciser restart operation and initializes the EXORciser. The MC6800 Microprocessing Unit is an 8-bit parallel device capable of addressing 64k bytes of memory. In addition, the MPU addresses its input and output devices as memory. The MPU also provides the EXORciser with 72 variable length instructions and the capability of responding to real time interrupt signals. With the exception of the EXORciser clock, the system restart, and the DBE signal (delayed); the MPU Module appears exactly like a MC6800 Microprocessing Unit with unlimited TTL drive capability.

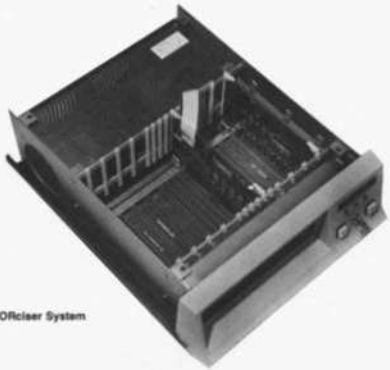


FIGURE 1-3. Basic EXORciser System

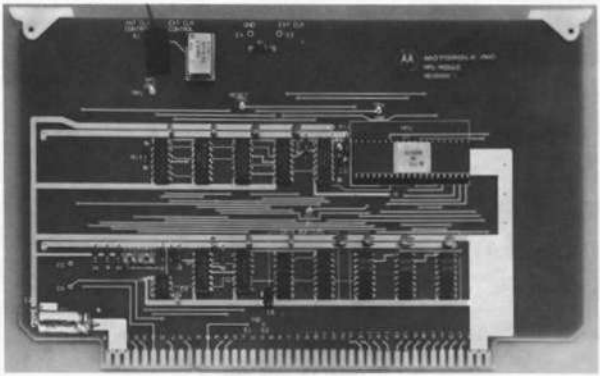


FIGURE 1-4. MPU Module

The Debug Module (Figure 1-5) provides the EXORciser with its capability to evaluate and debug the user's program. The MCM6830 ROM memory on this module contains the EXbug Firmware that provides the EXORciser with its unique software and hardware control features. The module also has a MCM6810 RAM memory to provide a scratch-pad memory to the EXbug Firmware. The EXbug Firmware enables the EXORciser to:

- Load data into the EXORciser
- Verify that the data in the EXORciser is valid
- Search a tape for a specific file
- Print the contents of the memory
- Punch (record) the contents of the memory on tape
- Perform the MAID (Motorola Active Interface Debur) functions

The MAID functions enable the EXORciser to:

- Examine and, if required, change the contents in a memory location
- Examine and, if required, change the data in an MPU register
- Calculate the offset in the relative addressing mode
- Insert, display, and remove breakpoints in a user's program
- Freerun or trace through a user's program under MAID control
- Stop the EXORciser on a specific program step in the user's program
- Provide an oscilloscope trigger pulse at a selected program step
- Search memory for a specific bit pattern
- Perform decimal-octal-hexadecimal conversions

The Debug Module working with the Baud Rate Module (Figure 1-6) provides the EXORciser with eight standard baud rates between 110 and 9600 Baud (refer to Table 1-1). These modules also interface the EXORciser with a TTY or RS-232C compatible terminal and provide a reader control signal for remote control of modified TTY terminals.

The Power Supply (Figure 1-7) provides the EXORciser with the +5VDC, +12VDC, and −12VDC power sources to support a full EXORciser rack of modules.

The chassis is capable of holding 14 plug-in modules. The Power Supply and Baud Rate Module are not plug-in modules but mounted directly to the chassis. Two versions of the chassis are available, the rack mounting version and the table-top version.

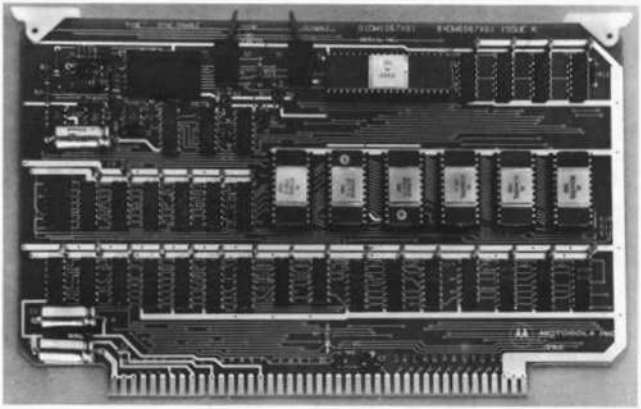


FIGURE 1-5. Debug Module

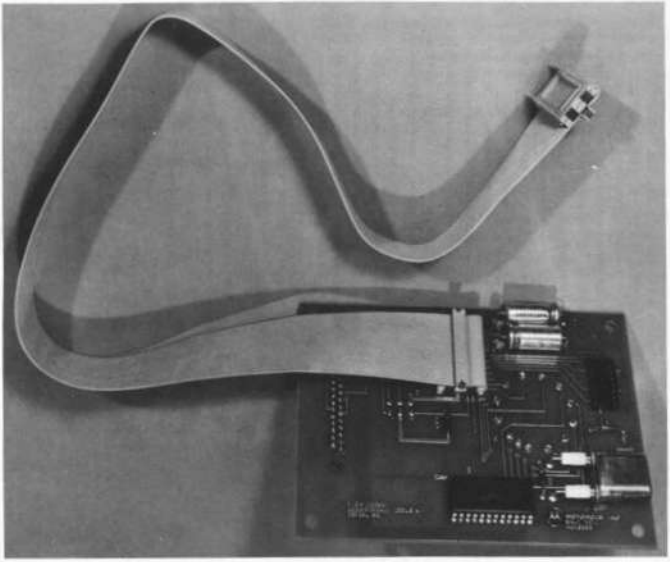


FIGURE 1-6. Baud Rate Module

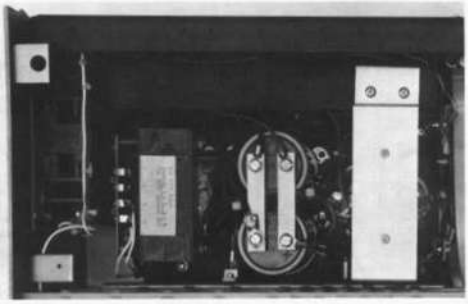


FIGURE 1-7. Power Supply

1-6 EXORciser OPTIONS

The EXORciser options provide the EXORciser with the flexibility of emulating the user's system. The optional modules provide the EXORciser with memory, peripheral interface capability, and provisions for constructing custom circuits. The EXORciser options are identified in Table 1-2.

1-6.1 UNIVERSAL WIREWRAP MODULE (FIGURE 1-8)

1-6.1.1 Features

The features of the Universal Wirewrap Module include:

- Standard pin spacing for 75 14-pin wirewrap sockets
- Standard size EXORciser plug-in module
- Permits user to build and incorporate his custom circuits into a system

1-6.1.2 Specifications

The Universal Wirewrap Module Specifications are identified in Table 1-3.

1-6.1.3 Description

The Universal Wirewrap Module permits the user to construct and incorporate his customize circuits into a M6800 Microcomputer System. This module is the standard size EXORciser plug-in module. The power bus and ground bus printed wiring are incorporated in the module. Also, the module has standard pin spacing for 75 14-pin wirewrap sockets and two 50-pin flatribbon cable connectors.

1-6.2 EXTENDER MODULE (FIGURE 1-9)

1-6.2.1 Features

The features of the Extender Module include:

- Extending any EXORciser plug-in module for servicing
- Interfacing with all EXORciser plug-in modules
- Providing clip-on test point for all EXORciser bus signals

1-6.2.2 Specifications

The Specifications of the Extender Module are identified in Table 1-4.

1-6.2.3 Description

The Extender Module enables the user to extend any EXORciser plug-in module for servicing, testing, troubleshooting, and debugging. This module provides clip-on test points for the EXORciser bus.

1-6.3 FLATRIBBON INTERCONNECT CABLE

The Flatribbon Interconnect Cable permits the user to connect an Input/Output Module to a peripheral device. One end of this cable is terminated with a 50-pin flatribbon connector and the other end is unterminated. Each I/O Module has provisions for two cables.

1-6.4 RACKMOUNTING KIT

The Rackmounting Kit allows the user to mount his EXORciser in a standard 19-inch rack.

1-6.5 TABLE TOP KIT

The Table Top Kit allows the user to use his EXORciser on a bench top or similar location.

TABLE 1-2. EXORciser OPTIONS

OPTION	PART NUMBER
2K Static RAM Module	MEX 6812-1
Input/Output Module	MEX 6820
Universal Wirewrap Module	MEX 68WW
Extender Module	MEX 68XT
Flatribbon Interconnect Cable	MEX 68IC
Rack Mounting Kit	MEX 68RK
Table Top Kit	MEX 68TT
8K Dynamic RAM Module	MEX 6815-1
Resident Software	
Option 1 • Assembler • Editor • 8K Dynamic RAM Module	M68XAE681.18
Option 2 • Assembler • Editor • Four 2K Static RAM Modules	M68XAE681.12
Option 3 • Assembler • Editor	M68XAE681.10

TABLE 1-3. Universal Wirewrap Module Specifications

CHARACTERISTICS	SPECIFICATIONS
Physical Characteristics	
Length	9.75 in.
Height	5.75 in.
Thickness	0.062 in.

TABLE 1-4. Extender Module Specifications

CHARACTERISTICS	SPECIFICATIONS
Physical Characteristics	
Length	9.75 in.
Height	9.00 in.
Thickness	0.062 in.

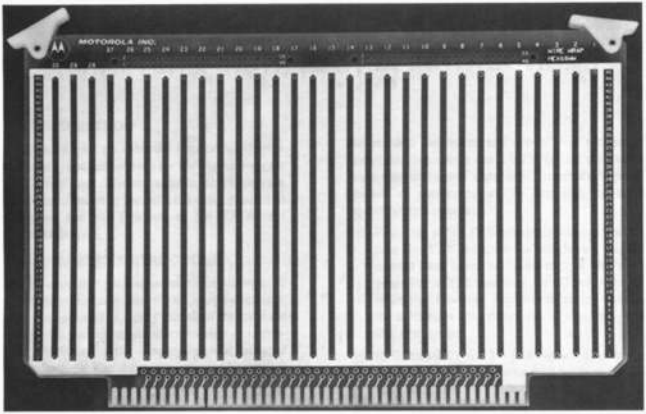


FIGURE 1-8. Universal Wirewrap Module

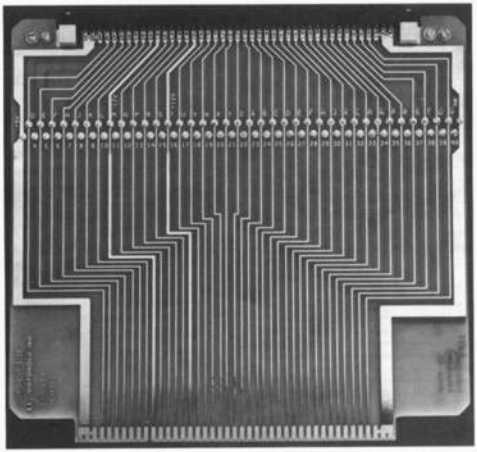


FIGURE 1-9. Extender Module

CHAPTER 2 INSTALLATION INSTRUCTIONS

2-1 INTRODUCTION

This chapter provides the unpacking, inspection, installation, and preparation for use instructions for the M6800 EXORciser. Also included in this chapter are instructions for connecting the selected data terminal to the EXORciser and the function of the EXORciser switches and indicators.

2-2 UNPACKING INSTRUCTIONS

No special unpacking instructions are required for the EXORciser. Unpack from the shipping carton and, referring to the packing list, verify that all of the parts are present. Save the packing material for storage or reshipment of the EXORciser. If the shipping carton is damaged upon receipt, request that the carrier's agent be present while the module is being unpacked and inspected.

2-3 INSPECTION

The EXORciser should be inspected upon receipt and at periodic intervals. Inspect the modules for broken, damaged, or missing parts and the printed circuit boards for physical damage. Inspect the connectors for damaged, bent, or missing pins. Inspect the cables for damaged insulation and broken wires.

2-4 PREPARATION FOR USE

The following paragraphs discuss connecting the selected data terminal to the EXORciser and preparing the EXORciser to operate with a 110 or 220 VAC source. The preparation of the optional modules is discussed in their respective supplements to this manual.

2-4.1 DATA TERMINAL CONNECTIONS

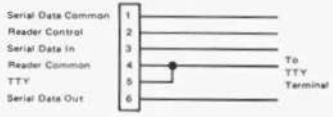
The user has the option of interfacing the EXORciser with an ASR33 Teletypewriter (TTY) or an RS-232C terminal device. The TTY data terminal shall be configured for automatic reader/punch control, full-duplex operation, and 20mA neutral current loop operation. The RS-232C terminal device should be capable of 110 to 9600 baud operation and incorporate automatic reader/punch (record) control. Paragraph 2-4.1.1 discusses the TTY terminal preparation and interconnections to the terminal and Paragraph 2-4.1.2 discusses the RS-232C interconnections.

2-4.1.1 TTY Compatible Terminal Preparation and Interface Connections

The EXORciser is capable of working with a TTY not having automatic reader/punch control capability; however, this is a long and tedious operation where the user has to manually turn his terminal on and off as required. Also, this terminal is not capable of supporting the resident assembler and editor. It is recommended, therefore, that a user use a TTY device with automatic reader/punch control capability. Appendix B presents a method of converting a TTY terminal for automatic reader/control operation.

(a) Referring to the TTY terminals documentation, prepare the TTY terminal for full-duplex and 20 mA current loop operation. Appendix B also discusses preparing a TTY terminal for 20 mA current loop operation; however, it is recommended that you use the terminal's documentation, if available, to make this conversion.

(b) Referring to Figure 2-1 and Table 2-1, construct the interconnection cable to be used between the EXORciser and the TTY terminal. Use the connector supplied with the EXORciser.

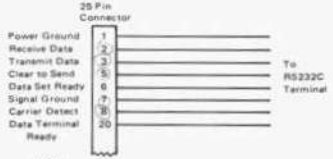


NOTE:
Use Cinch DE-9P or Equivalent Connector

FIGURE 2-1. TTY/EXORciser Interconnection Cable

2-4.1.2 RS-232C Compatible Terminal Interface Connections

Referring to Figure 2-2 and Table 2-2, construct the interconnect cable to be used between the EXORciser and the RS-232C terminal. Use the 25-pin connector supplied with the EXORciser.



NOTE:
1. All other pins are not used.
2. Use cinch DB-25P or equivalent connector

FIGURE 2-2. RS-232C/EXORciser Interconnection Cable

TABLE 2-1. TTY/EXORciser Interconnections

PIN NUMBER	SIGNAL MNEMONIC	SIGNAL NAME AND DESCRIPTION
1	SERIAL DATA COMMON	SERIAL DATA COMMON — This −12 VDC signal provides the signal return for a TTY data terminal.
2	READER* CONTROL	READER CONTROL — This signal provides the control to operate the paper tape reader on a modified TTY data terminal under MPU control.
3	SERIAL* DATA IN	SERIAL DATA IN — This line accepts the input from a TTY data terminal.
4	READER COMMON	READER COMMON — This line is connected to EXORciser ground. This pin is also connected to pin 5 in the cable.
5	TTY	TTY — This line indicates when a TTY device is connected to the EXORciser. This pin is connected to pin 4 in the cable.
6	SERIAL* DATA OUT	SERIAL DATA OUT — This line transfers data to a TTY data terminal.

*The EXORciser provides the necessary drive for the TTY data transfer.

2-4.2 POWER SUPPLY CONNECTIONS

The EXORciser power supply may be configured to accept 95 to 135 VAC, (110 VAC) or 205 to 250 VAC (220 VAC) input voltage with an input frequency of 47 to 420 Hz. This power supply provides the EXORciser with +5 VDC @ 15A, +12 VDC @ 2.5A, and −12 VDC @ 1.5A. Paragraph 2-4.2.1 discusses configuring the power supply for a 110 VAC input voltage and Paragraph 2-4.2.2 discusses configuring for a 220 VAC input voltage.

2-4.2.1 Power Supply 110 VAC Voltage Connections (Refer to Figures 2-3 and 2-4)

(a) Connect one of the power lines to pins 1 and 2 of the transformer terminal board

(b) Connect the other side of the power line to pins 3 and 4 of the transformer terminal board.

(c) Connect the fans as illustrated in Figure 2-4.

2-4.2.2 Power Supply 220 VAC Voltage Connections (Refer to Figures 2-3 and 2-4)

(a) Connect one of the power lines to pin 2 of the transformer power supply.

(b) Connect the other power line to pin 3 of the transformer power supply.

(c) Connect a jumper between pins 1 and 4 of the transformer power supply.

(d) Connect the fans as illustrated in Figure 2-4.

2-5 INSTALLATION INSTRUCTIONS

The user has the option in purchasing his EXORciser of purchasing the desk top version or the rack mounting version. Paragraph 2-5.1 discusses setting up the table top version while Paragraph 2-5.2 discusses installing the rack mounted version. Paragraph 2-5.3 discusses installing the printed circuit module's into the EXORciser.

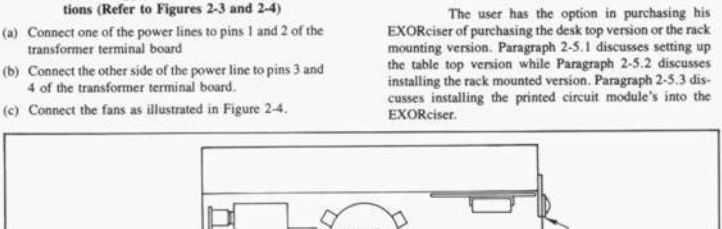


FIGURE 2-3. EXORciser Power Supply

TABLE 2-2. RS-232C/EXORciser Interconnections

PIN NUMBER	SIGNAL MNEMONIC	SIGNAL NAME AND DESCRIPTION
1	POWER GND	POWER GROUND — Common for the −12 volt source. This line provides a safety ground connection to the RS-232C compatible terminal.
2	RECEIVE DATA	RECEIVE DATA — This line accepts the input from an RS-232C compatible terminal.
3	TRANSMIT DATA	TRANSMIT DATA — This line transfers data to an RS-232C compatible terminal.
4		Not used
5	CLEAR TO SEND	CLEAR TO SEND — This line is a high level when an RS-232C data terminal is connected to the EXORciser and informs the terminal to transfer data.
6	DATA SET READY	DATA SET READY — This line is a high level when a RS-232C data terminal is connected to the EXORciser and indicates to the terminal that the EXORciser is operating.
7	SIGNAL GND	SIGNAL GROUND — This line provides a common signal connection to the RS-232C data terminal.
8	CARRIER DETECT	CARRIER DETECT — This line is a high level when a RS-232C data terminal is conneted to the EXORciser and indicates that the EXORciser has detected the carrier signal.
9		Not used
10		Not used
11		Not used
12		Not used
13		Not used
14		Not used
15		Not used
16		Not used
17		Not used
18		Not used
19		Not used
20	DATA READY	DATA TERMINAL READY — This line from the RS-232C data terminal indicates that the data terminal is ready.
21		Not used
22		Not used
23		Not used
24		Not used
25		Not used

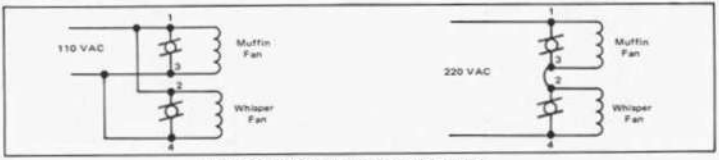


FIGURE 2-4. EXORciser Power Distribution Drawing

2-5.1 RACK MOUNTING INSTALLATION INSTRUCTIONS

In selecting the site to mount your EXORciser, make sure you provide adequate space in front of the rack to allow the EXORciser to be fully extended from the rack. Also, mount the EXORciser so that you have easy access to the EXORciser modules when the EXORciser is extended from the rack.

(a) Mount the EXORciser in a standard 19" retma rack.

(b) With the EXORciser PWR switch OFF, connect the EXORciser to the selected power source.

(c) Connect the selected data terminal to the EXORciser, using the cable constructed in Paragraph 2-4. A TTY data terminal is connected to EXORciser connector J3 while a RS-232C data terminal is connected to EXORciser connector J2.

(d) Install the optional modules in accordance with Paragraph 2-5.3.

2-5.2 TABLE TOP INSTALLATION INSTRUCTION

In selecting a site to place your EXORciser, select a site where you have easy access to the EXORciser front panel and to the EXORciser modules.

(a) Place the EXORciser in the selected location.

(b) With the EXORciser PWR switch OFF, connect the EXORciser to the selected power source.

(c) Connect the selected data terminal to the EXORciser using the cable constructed in Paragraph 2-4. A TTY data terminal is connected to EXORciser connector J3 and a RS-232C device is connected to EXORciser connector J2.

(d) Install the optional modules in accordance with Paragraph 2-5.3

2-5.3 PRINTED CIRCUIT BOARD INSTALLATION

The installation of the optional EXORciser modules is discussed in their supplements to this manual. Since each module is offset as well as keyed, there is little chance of installing the modules in backwards. The EXORciser bus permits the user to install the modules in any of the 14 card slots; however, modules having wire wrap sockets require additional spacing between card slots. The 14 card slots are illustrated in Figure 2-5. Install the modules as follows:

(b) If the EXORciser is mounted in a rack, pull the EXORciser out on its extenders to gain full access to the dust cover.

(b) Remove the dust cover from the EXORciser. The rack mounted dust cover is held in place by four screw-lock fasteners while the table top dust cover is held in place by tension clips.

(c) Referring to the module supplements, install the optional modules into their selected card slots. (It is assumed that the Debug and MPU Modules are already installed in their selected card slots.)

NOTE

It is recommended that the dust covers be placed on the EXORciser when you are not working on the EXORciser.

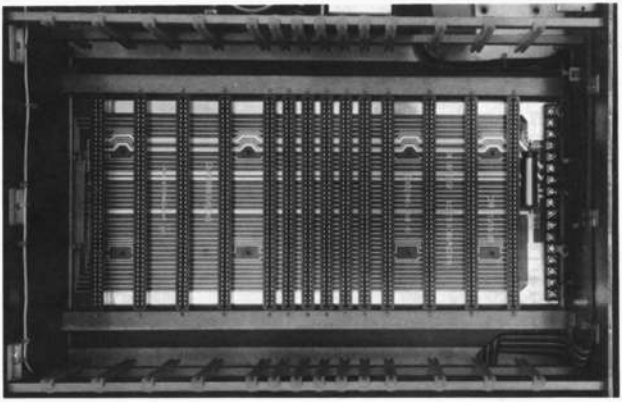


FIGURE 2-5. EXORciser Card Rack

2-6 SWITCHES AND INDICATORS

The EXORciser switches and controls are divided into three categories, those on the EXORciser chassis, those used on the Debug and MPU Modules, and those used on the optional modules. Figure 2-6 and Table 2-3 identify the location and function of the switches and indicators on the EXORciser while Figures 2-7 and 2-8 along with Tables 2-4 and 2-5 identify the location and function of the MPU and Debug Module switches. The switches on the optional modules are identified in their respective supplements.

Fig. 2-6 Front View

Fig. 2-6 Back View

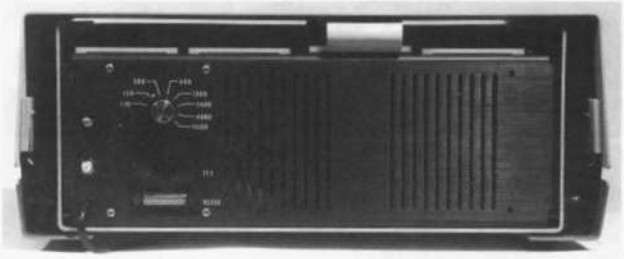


FIGURE 2-6. EXORciser Switches and Indicators

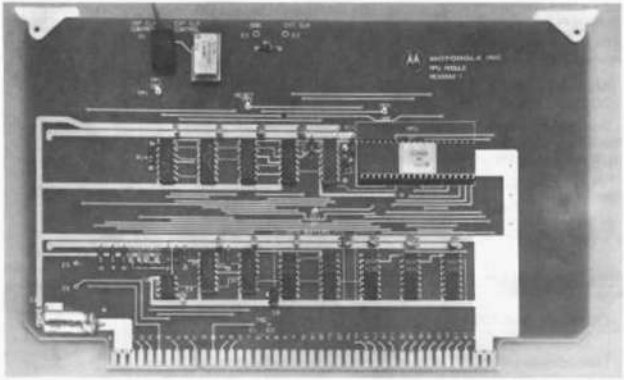


FIGURE 2-7. MPU Module Switch Locations

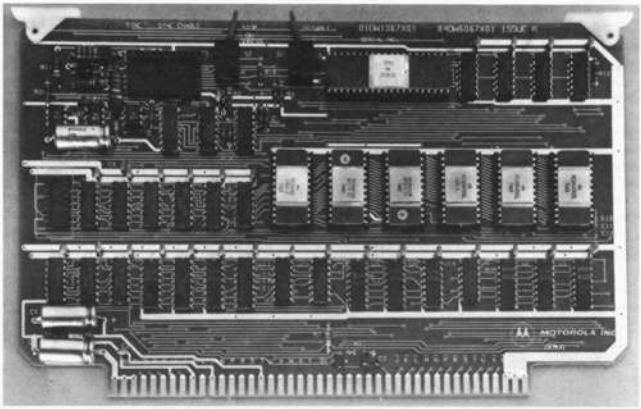


FIGURE 2-8. Debug Module Switch Locations

2-7 TABLE TOP TO RACK MOUNTING CONVERSION

The following procedures discuss converting the EXORciser from its table top version to its rack mounted version. Figure 2-9 depicts an exploded view of the table top kit and Figure 2-10 depicts an exploded view of the rack mounting kit.

(a) Remove the Table Top Kit from the EXORciser in the reverse order of the index numbers in Figure 2-9.

(b) Install the Rack Mounting Kit onto the EXORciser in the sequence of the index numbers in Figure 2-10.

2-8 RACK MOUNTING TO TABLE TOP CONVERSION

The following procedures discuss converting the EXORciser from its rack mounting version to its table top version. Figure 2-9 depicts an exploded view of the table top kit and Figure 2-10 depicts an exploded view of the rack mounting kit.

(a) Remove the rack mounting kit from the EXORciser in the reverse order of the index numbers in Figure 2-10.

(b) Install the table top kit onto the EXORciser in the sequence of the index numbers in Figure 2-9.

2-9 EXORciser BUS SIGNAL DESIGNATIONS

Table 2-6 identifies each of the signals on the EXORciser bus, the pin connection, the signal's mnemonics, and the signal's characteristics.

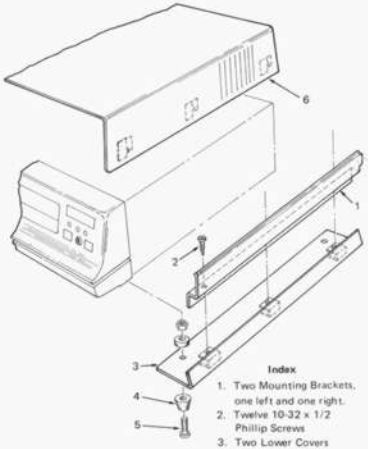


FIGURE 2-9. Table Top Kit

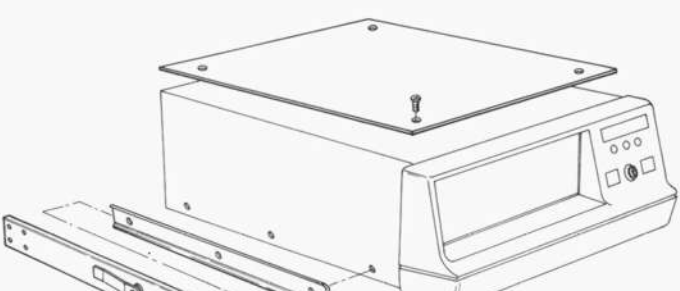


FIGURE 2-10. Rack Mounting Kit

TABLE 2-3. EXORciser Switches and Indicators

SWITCH/ INDICATOR	FUNCTION
RUN Indicator	The RUN indicator is on when the EXORciser is running the user's program and is not on when the EXORciser is in EXbug (Upper 4K of memory).
PWR Indicator	The PWR (PoWeR) indicator indicates the status of the EXORciser PWR keyswitch. The PWR indicator is ON when the PWR switch is ON.
AUX Indicator	The AUX (AUXiliary) indicator is not used at this time. (Wired in cable harness and available on the Baud Rate Module).
RESTART Switch	The RESTART pushbutton switch, when actuated, resets and initializes the EXORciser. The EXORciser on completing its initialization routine, goes to its EXbug control program.
PWR	The PWR (PoWeR) key switch, when on, applies primary power to the EXORciser.
ABORT	The ABORT pushbutton switch, when actuated, aborts or exits the EXORciser from the routine it is performing and returns the EXORciser to its EXbug control program.
BAUD RATE (Switch)	The BAUD RATE switch allows the user to select the data transfer rate between the EXORciser and the data terminal.

TABLE 2-4. MPU Module Switch Functions

SWITCH NAME	SWITCH FUNCTION
CLOCK Switch	This switch selects either the EXORciser's 1 MHz internal clock or an external clock. The external clock input is located on top of the MPU Model and provides a 50 ohm load.

TABLE 2-5. Debug Module Switch Functions

SWITCH NAME	SWITCH FUNCTION
EXBUG Switch	The EXbug Switch in the ON position, enables the EXORciser to execute the EXbug Firmware.
SCOPE/STOP-ON-ADDRESS	The SCOPE/STOP-ON-ADDRESS switch, in the SCOPE position, provides a scope trigger pulse on a selected address. This switch, in the STOP-ON-ADDRESS position, instructs the EXORciser to stop on a selected memory address. The EXORciser will stop the next program instruction.

TABLE 2-6. EXORciser BUS Signals

PIN NUMBER	SIGNAL MNEMONIC	SIGNAL NAME AND DESCRIPTION
A	+5 VDC	+5 VDC used for the module's logic circuits.
B	+5 VDC	+5 VDC used for the module's logic circuits.
C	+5 VDC	+5 VDC used for the module's logic circuits.
D	$\overline{IRQ}$	INTERRUPT REQUEST (IRQ) — This signal requests that a MPU interrupt sequence be generated within the machine. The MPU will wait until it completes the current instruction that it is executing before it recognizes the request. At that time, if the interrupt mask bit in the MPU condition code register is not set (interrupt masked), the MPU will begin the interrupt sequence.
E	$\overline{NMI}$	NON-MASKABLE INTERRUPT (NMI) — This signal requests that a non-maskable interrupt be generated within the machine. The MPU will wait until it completes the current instruction that it is executing before it recognizes the NMI signal. At that time, the MPU will begin its non-maskable interrupt sequence. The EXORciser EXbug Firmware uses the non-maskable interrupt for its Abort, Run-One-Instruction, and Stop-On-Address functions.
F	VMA	VALID MEMORY ADDRESS (VMA) — This output indicates to the Debug Module that there is a valid memory address on the address bus. This signal is not three-state.
H	GND	GROUND for ±12 VDC
J	$\phi 2$	Phase 2 ($\phi 2$) clock signal
K	GND	GROUND for ±12 VDC
L	MEMCLK	MEMORY CLOCK (MEMCLK)—This signal is the basic clock frequency used by the MPU Module to generate its $\phi 1$ and $\phi 2$ clock signals.
M	−12 VDC	−12 VDC for use with user's custom designed interface circuitry.
N	$\overline{TSC}$	THREE-STATE CONTROL (TSC) — This input, when high, places all of the address lines in their off or high impedance state. The VMA and BA signals will be forced low. The data bus is not affected by the TSC input.

P	BA	BUS AVAILABLE (BA) — This bus available output signal will normally be a low level; when activated, it will go high indicating that the MPU has stopped and that the address bus is available. This will occur if the Go/Halt line is in the Halt (low) state or the MPU is in the WAIT state as a result of executing a WAIT instruction. At such time, all the MPU three-state output drivers will go to their off state and other outputs to their normally inactive state. A maskable interrupt, or actuating the RESTART or ABORT switches, removes the MPU from the WAIT state.
R	MEMRDY	MEMORY READY (MEMRDY) — This signal, when present, initiates a memory refresh operation. At this time the MPU Module is inhibited from generating its $\phi 1$ and $\phi 2$ clock signals.
S	REFCLK	REFRESH CLOCK (REFCLK) — This signal is generated by the dynamic memory module being used as the master refresh module. This signal is used to initiated a memory operation on the dynamic modules used as slave refresh modules.
T	+12 VDC	+12 VDC for use with user's custom designed interface circuitry.
U	BAT +12	BATTERY +12 VOLTS (BAT +12) — This line in normal EXORciser operation is +12 VDC from the EXORciser power supply. If the EXORciser is using battery backup and the power supply is turned off or a power-fail should occur, the power backup places +12 VDC on this line.
V	$\overline{\text{STDBY}}$	STAND BY (STDBY) — This line is a low level during a power-fail condition and a high level during normal EXORciser operation.
W		Not used
X		Not used
Y		Not used
Z		Not used
$\overline{\text{A}}$		Not used
$\overline{\text{B}}$	GND	GROUND
$\overline{\text{C}}$		Not used
$\overline{\text{D}}$		Not used
$\overline{\text{E}}$	(BA)	BUS AVAILABLE (BA) — This line is present only when an Evaluation Module is placed in the EXORciser chassis. Signal is same as BA on pin P1-P.
$\overline{\text{F}}$	(G/H)	GO-HALT — This line is used only when an Evaluation Module is placed in the EXORciser chassis. Signal is same as G/H on P1-4.
$\overline{\text{H}}$	$\overline{\text{D3}}$	DATA BUS (D3) — This bi-directional line, when enabled, provides a two-way data transfer between the MPU Module and all other plug-in modules in the EXORciser. The data bus drivers on these module are in their off or high-impedances state except when one of these modules is selected during a memory read operation.
$\overline{\text{J}}$	$\overline{\text{D7}}$	DATA BUS ($\overline{\text{D7}}$) — Same as $\overline{\text{D3}}$ on P1-$\overline{\text{H}}$.
$\overline{\text{K}}$	$\overline{\text{D2}}$	DATA BUS ($\overline{\text{D2}}$) — Same as $\overline{\text{D3}}$ on P1-$\overline{\text{H}}$.
$\overline{\text{L}}$	$\overline{\text{D6}}$	DATA BUS ($\overline{\text{D6}}$) — Same as $\overline{\text{D3}}$ on P1-$\overline{\text{H}}$.
$\overline{\text{M}}$	A14	ADDRESS BUS (A14) — This address line, when enabled, transfers the MPU program counter output to add the plug-in modules in the EXORciser.
$\overline{\text{N}}$	A13	ADDRESS BUS (A13) — Same as A14 on P1-$\overline{\text{M}}$.
$\overline{\text{P}}$	A10	ADDRESS BUS (A10) — Same as A14 on P1-$\overline{\text{M}}$.

$\overline{R}$	A9	ADDRESS BUS (A9) — Same as A14 on P1-$\overline{M}$.
$\overline{S}$	A6	ADDRESS BUS (A6) — Same as A14 on P1-$\overline{M}$.
$\overline{T}$	A5	ADDRESS BUS (A5) — Same as A14 on P1-$\overline{M}$.
$\overline{U}$	A2	ADDRESS BUS (A2) — Same as A14 on P1-$\overline{M}$.
$\overline{V}$	A1	ADDRESS BUS (A1) — Same as A14 on P1-$\overline{M}$.
$\overline{W}$	GND	GROUND
$\overline{X}$	GND	GROUND
$\overline{Y}$	GND	GROUND
1	+5 VDC	+5 VDC used for the module's logic circuits.
2	+5 BDC	+5 VDC used for the module's logic circuits.
3	+5 VDC	+5 VDC used for the module's logic circuits.
4	G/H	GO/HALT (G/H) — When this input is in the high state, the MPU will fetch the instruction addressed by the program counter and start execution. When low, all activity in the MPU will be halted. This input is level sensitive. In the halt mode, the MPU will stop at the end of an instruction, bus available will be at a high level, valid memory address will be at low level, and all other three-state lines will be in their off or high impedance state. The halt line must go low with the leading edge of the phase one clock (ϕ1) to insure single instruction operation. If the halt line does not go low with the leading edge of the phase one clock, one or two instruction operations may result, depending on when the halt line goes low relative to the phasing of the clock.
5	$\overline{\text{RESET}}$	RESET — This signal is used to start the MC6800 MPU and reset the EXORciser from a power down condition or when the EXORciser RESTART switch is actuated.
6	R/W	READ/WRITE (R/W) — This MPU output signal indicates to the EXORciser plug-in modules whether the MC6800 MPU is performing a memory read (high) or write (low) operation. The normal standby state of this signal is read (high). Also, when the MC6800 is halted, this signal will be in the read state.
7	ϕ2	Phase 2 (ϕ2) clock signal
8	GND	GROUND for ±12 VDC
9	GND	GROUND for ±12 VDC
10	VUA	VALID USER'S ADDRESS (VUA) — This signal, when high, indicates that the address on the address bus is valid and the EXORciser is not addressing its EXbug program.
11	−12 VDC	−12 VDC for use with the user's custom designed interface circuitry.
12	$\overline{\text{REFREQ}}$	REFRESH REQUEST (REFREQ) — This signal, when low, initiates a memory refresh cycle of the dynamic memory modules. During the refresh operation the MPU Module is inhibited from generating its ϕ1 and ϕ2 clock signals.
13	REF GRANT	REFRESH GRANT (REF GRANT) — This signal, when high, instructs the dynamic memory modules to refresh their memories.
14		Not used
15		Not used
16	+12 VDC	+12 VDC for use with the user's custom designed interface circuitry.
17	BAT +12	BATTERY +12 VOLTS (BAT +12) — Same sd BAT +12 on P1-U.
18	$\overline{\text{(TSC)}}$	Three-State Control (TSC) — This line is present only when an Evaluation Module is placed in the EXORciser chassis. Signal is same as TSC on P1-N.
19		Not used

20		Not used
21	AC OFF	AC OFF — In a system using battery backup, this signal indicates when ac power is removed from the primary of the power supply transformer.
22		Not used
23		Not used
24	GND	GROUND
25		Not used
26		Not used
27		Not used
28		Not used
29	$\overline{D1}$	DATA BUS ($\overline{D1}$) — Same as D3 on P1-$\overline{H}$
30	$\overline{D5}$	DATA BUS ($\overline{D5}$) — Same as D3 on P1-$\overline{H}$
31	$\overline{D0}$	DATA BUS ($\overline{D0}$) — Same as D3 on P1-$\overline{H}$
32	$\overline{D4}$	DATA BUS ($\overline{D4}$) — Same as D3 on P1-$\overline{H}$
33	A15	ADDRESS BUS (A15) — Same as A14 on P1-$\overline{M}$.
34	A12	ADDRESS BUS (A12) — Same as A14 on P1-$\overline{M}$.
35	A11	ADDRESS BUS (A11) — Same as A14 on P1-$\overline{M}$.
36	A6	ADDRESS BUS (A8) — Same as A14 on P1-$\overline{M}$.
37	A7	ADDRESS BUS (A7) — Same as A14 on P1-$\overline{M}$.
38	A4	ADDRESS BUS (A4) — Same as A14 on P1-$\overline{M}$.
39	A3	ADDRESS BUS (A3) — Same as A14 on P1-$\overline{M}$.
40	A0	ADDRESS BUS (A0) — Same as A14 on P1-$\overline{M}$.
41	GND	GROUND
42	GND	GROUND
43	GND	GROUND

CHAPTER 3 M6800 SYSTEM EMULATION

3-1 INTRODUCTION

The M6800 EXORciser is a system development tool used in the design and development of M6800 Microprocessor Systems. It is assumed in this chapter that the user is designing his system as described in the M6800 Microprocessor Applications Manual and is preparing his programs in accordance with the M6800 Microprocessor Programming Manual. This chapter reviews in general terms developing the hardware, preparing the software, and the place of the EXORciser in this system development. This chapter also presents preparing the EXORciser to emulate (functionally duplicate) and debug the user's system.

This chapter discusses using the optional EXORciser module in general terms. Refer to the optional module's supplements to this user's guide as required to assist you in preparing your EXORciser to emulate your system.

3-2 M6800 EXORciser AND SYSTEM DEVELOPMENT

To better understand using the EXORciser in the design and development of a M6800 Microprocessor system, let us first review the procedure followed by a typical engineer in developing a microprocessor system and, how the EXORciser will simplify the design effort. It should be noted that the system design procedures presented were developed for discussion purposes only.

3-2.1 DEVELOPING MICROCOMPUTER SYSTEMS

The following procedures are intended to illustrate how an engineer might develop a typical microcomputer system. The enginer:

(1) Defines his system. In this definition he determines the software and hardware functions to be performed and the interfacing requirements.

(2) Designs and constructs a hardware prototype of his system and at the same time,

(3) Prepares his software programs around the hardware to accomplish the intended function.

(4) Combines the software with his hardware and debugs both his hardware and his software until he has a working system.

(5) Finalizes his system design and his software and builds a preproduction model of his system.

(6) Combines the preproduction hardware with the system software and makes any necessary hardware and software adjustments.

(7) Extensively tests and evaluates his system in an actual working environment. The user's program may be stored in PROMs.

(8) Builds the production hardware of his system and has his ROMs made.

(9) Combines the production hardware with the system software (in ROM) and makes the final system adjustments.

(10) Releases his system to production.

3-2.2 M6800 EXORciser IN SYSTEM DEVELOPMENT

The EXORciser, through its ability to emulate a user's system hardware and debug a user's system, greatly reduces the time required for an engineer to construct a prototype of and debug his system. The engineer, rather than designing and building a prototype of his system, sets up the EXORciser to functionally represent his system. He may however, have to construct special circuitry to interface the EXORciser to his process or peripheral device. Now the engineer has developed his system using proven hardware with a minimum amount of design and construction time.

The EXORciser provides the programmer with a functional system on which to develop his system software. The advantages of the EXORciser to the programmer are that the EXORciser permits him to assemble, edit, and modify his program on the actual hardware and to evaluate his programs using the system input/output capabilities.

The EXORciser's EXbug Firmware provides the engineer and the programmer with a system development program capability for debugging both the system hardware and the system software. This EXbug system development program is hardware compatible and enables the user to:

- Load his program into the EXORciser
- Verify that the program was correctly loaded
- Search a tape for a specific file
- Print the contents of the memory
- Punch (record) the contents of the memory on tape
- Debug his system hardware and software using MAID (Motorola Active Interface Debug) functions.

These MAID functions permit the user to debug his system through:

- Examining and, if required, changing the contents in a memory location

- Examining and, if required, changing the contents in the MPU registers
- Stopping the EXORciser on a specific memory address in a user's program
- Providing an oscilloscope trigger pulse at a selected memory address
- Calculating the offset in the relative addressing mode
- Inserting, displaying, and removing breakpoints in a user's program
- Freerunning or tracing through a user's program under development
- Searching the memory for a specific bit pattern
- Performing decimal-octal-hexadecimal conversions.

If the user were to develop his system without the EXORciser, he would be required to develop or purchase his own system development program using at least part of the EXbug functions.

Now, let us review the system development process using the EXORciser. The engineer:

(1) Defines his system as before. In this definition he determines the software and hardware functions to be performed.

(2) Sets up the EXORciser to emulate his system hardware. He will, if required, also design and build any special hardware interface circuitry.

(3) Prepares his software programs around the hardware to accomplish the intended function.

(4) Loads software into EXORciser and, using the EXbug Firmware, debugs both the hardware and the software until he has a working system.

(5) Designs and builds a preproduction model of his system compatible with the EXORciser's bus.

(6) Combines the preproduction hardware with the user's system software and, using the EXORciser's EXbug Firmware, debugs and makes any necessary system hardware and software adjustments.

(7) Extensively tests and evaluates his system in an actual working environment. At this point the program may be stored in the PROMs.

(8) Builds the production hardware of his system and has his ROMs made. This system should be compatible with the EXORciser bus if the EXORciser is to be used in debugging the system.

(9) Combines the production hardware with the system software and makes the final adjustments to his system. He again may use the EXORciser in evaluating his system.

(10) Releases his system to production

3-3 HARDWARE PREPARATION

This paragraph discusses preparing the EXORciser to emulate (functional represent) the user's system. This paragraph discusses preparing the EXORciser's optional modules in general terms. Refer to the options modules' supplements as required to prepare these modules. Prepare the EXORciser as follows:

(a) Construct any required special circuitry, i.e. special interface circuitry.

(b) Using the procedures in Chapter 2, install the EXORciser modules in the selected EXORciser card slots.

(c) Determine whether you are going to use the EXORciser clock or an external clock in your system. Set the clock switch S1 on the MPU Module accordingly.

(d) If you are using an external clock, determine the clock frequency to be used in your system.

(e) Connect the external clock to the MPU Module.

(f) Connect the EXORciser to the user's process or peripheral device.

(g) Set the base memory addresses on the EXORciser's memory and peripheral interface modules.

3-4 PROGRAMMING PREPARATION

The following paragraphs are provided to assist you in preparing your program. These paragraphs are a supplement to the M6800 Microcomputer Programming Manual and are intended as a guide. Also, these paragraphs discuss the optional modules only in general terms. Each optional modules' unique programming considerations are discussed in their supplements to this manual. Refer to the Programming Manual and these supplements as required in preparing the user's program.

Also included in the following paragraphs are the EXORciser's programming constraints, the programming characteristics of the EXORciser, memory organization, special programming considerations, and the suggested procedures used in program preparation.

3-4.1 PROGRAMMING CONSTRAINTS

The EXORciser places the following constraints on the user while he is using the EXbug Firmware.

- He can not use memory locations F000 through FFFF.
- The EXbug Firmware uses the MC6800 MPU's NMI (Non-Maskable-Interrupt) capability.
- The EXbug Firmware uses the MC6800 MPU's SWI (SoftWare-Interrupt) capability.

3-4.1.1 EXbug Firmware

The DEBUG switch on the Debug Module allows the user to use or not use the EXbug Firmware. This Firmware precludes the user from using memory locations F000 through FFFF. Now, the EXORciser may be operating with partically coded user's systems (not using all 16 address lines) and programs using addresses between F000 and FFFF. To prevent reading duplicate addressing, the EXORciser, on determining that it is reading an EXbug address, inhibits the reading of a user's program.

If you have a portion of your program above F000 and wish to debug it using EXbug, it is recommended that you assemble this program at another memory location, debug it, and, upon completion, reassemble the program at its original address.

3-4.1.2 SWI (Software Interrupt) Capability

The EXORciser EXbug Firmware uses the MC6800 MPU's SWI capability to insert breakpoints into the user's program (Paragraph 4-14). You can use the EXORciser to evaluate a user's software interrupt at the expense of losing the EXORciser's breakpoint function capability. If you wish to evaluate a software interrupt routine, proceed as follows:

(a) Using the MAID Memory Change Function (Paragraphs 4-9 and 4-11), change the interrupt vector in memory location FFFA (MSB) and FFFB (LSB) to the address of your software interrupt routine.

NOTE

The EXORciser, on the actuation of the RESTART or ABORT switch, enters its MPU cold-start power-up routine and loads addresses FFFA and FFFB with the breakpoint vector address. If you are evaluating your software interrupt routine and are required to actuate one of these switches, repeat step a.

(b) Initiate, run, and debug the user's software interrupt routine.

(c) After you have completed evaluating the software interrupt routine, press the ABORT switch, and restore the EXbug breakpoint function.

3-4.1.3 NMI (Non-Maskable Interrupt) Capability

The EXORciser EXbug Firmware uses the MC6800 MPU's NMI capability in its Abort function, Stop-on-Address function, and Run-One-Instruction feature. You can use the EXORciser to evaluate a user's non-maskable interrupt routine at the expense of losing the EXORciser's Abort, Stop-On-Address, and Run-One-Instruction capabilities. If you wish to evaluate a non-maskable interrupt routine, proceed as follows:

(a) Using the MAID Memory Change Function (Paragraphs 4-9 and 4-11), change the interrupt vector in memory locations FFFC (MSB) and FFFD (LSB) to the address of your non-maskable interrupt routine.

NOTE

The EXORciser, on the actuation of the RESTART switch, enters its MPU cold-start power-up routine and loads the addresses FFFC and FFFD with the EXbug non-maskable interrupt routine vector address. If you are evaluating your software interrupt routine and are required to actuate one of these switches, repeat step a.

(b) Initiate, run, and debug the user's non-maskable interrupt routine.

(c) After you have completed evaluating the non-maskable interrupt routine, press the RESTART switch and restore the EXbug NMI functions.

3-4.2 UNIQUE PROGRAMMING CONSIDERATIONS

The MC6800 Microprocessing Unit sets aside the top eight addresses of memory — FFF8 through FFFF — for its interrupt vectors as follows:

- IRQ uses addresses FFF8 and FFF9
- SWI uses addresses FFFA and FFFB
- NMI uses addresses FFFC and FFFD
- RESTART uses addresses FFFE and FFFF.

The user in designing his system, must make his top of memory addresses appear to the MC6800 MPU as addresses FFF8 through FFFF (refer to the M6800 Microprocessor Applications and Programming Manuals for details).

The EXbug Firmware assumes that the user is using 83FF as the top address in the user's program. That is, the EXbug Firmware assumes that the user is using only address lines 2^{15} and 2^{9} through 2^{0} in the user's system. Using this addressing configuration, the addresses 83FE and 83FF appear to the MC6800 MPU as addresses FFFE and FFFF (the MPU RESTART vector address). If you are using an address other than 83FF as the top of the user's system, you will have to enter the memory top of addresses as described in Paragraph 4-2. Also, each time the EXORciser RESTART switch is actuated, the EXORciser performs its cold-start power-up routine and initializes itself. If the user's system is using a top of memory address other than 83FF, you must reenter the user's system top of memory addresses in accordance with Paragraph 4-2.

3-4.3 MEMORY ASSIGNMENTS

The user, before he prepares his equipment and writes his program, must assign the memory locations to be used in his system. Figure 3-1 illustrates a typical EXORciser memory map. Prepare your map as follows:

(a) Construct a basic memory map of your system. On this map indicate that the EXORciser is using memory addresses F000 through FFFF.

(b) Assign the memory location for your MC6830 ROM's at the top of your memory. The top of memory of the ROMs must appear to the MPU as address FFFF.

(c) Assign the memory addresses for your MC6810 RAM's. It is recommended that you place the RAM at the bottom of memory to take advantage of direct addressing mode.

(d) If your system is using MC6820 Peripheral Interface Adapters (PIA) assign four addresses for each PIA as described in the M6800 Microprocessor Applications Manual.

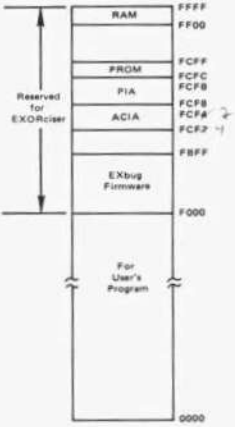


FIGURE 3-1. EXORciser Map

(e) If your system is using MC6850 Asynchronous Communication Interface Adapters (ACIA) assign two addresses for each ACIA as described in the M6800 Microprocessor Applications Manual.

(f) Determine the required address lines to be used in your system.

3-4.4 IRQ (INTERRUPT REQUEST) AND INTERRUPT PRIORITIES

In its initializing routine the EXORciser transfers the IRQ vector address from the user's program to memory locations FFF8 and FFF9. The EXORciser, through its bus, transfers IRQ (Interrupt ReQuest) inputs from a peripheral device to the MC6800 MPU. The MPU completes the instruction it is executing before it recognizes an IRQ input. Then, if the interrupt mask bit in the MPU's condition code register is not set, the EXORciser reads the vector address in memory locations FFF8 and FFF9 and jumps to the user's IRQ routine.

In setting up the user's system, you must determine whether you are going to use the MC6800 MPU's IRQ capability and, if you are, you must determine the interrupt command input level from the peripheral device to the EXORciser. Refer to the M6800 Microprocessor Applications Manual and M6800 Microprocessor Programming Manuals and the supplements to this user's guide to set up or inhibit the IRQ capabilities for the MC6820 PIA and MC6850 ACIA devices.

If you are using more than one peripheral device capable of interrupting the MC6800 MPU, you must prepare an interrupt polling routine to interrogate each of these devices and determine the device initiating the interrupt. This polling routine, on determining the IRQ initiated device, causes the MPU to jump to the appropriate service routine.

In preparing the interrogation routine, you must assign priorities to the devices. Assign priorities to these devices by assigning their position in the interrogation polling routine. The device that is interrogated first has the highest priority and the device interrogated last has the lowest priority.

3-4.5 MC6820 PIA PROGRAMMING INFORMATION

The MC6800 MPU addresses the MC6820 PIA as if it were memory. Therefore, all commands to the PIA are executed by the MPU as memory reference instructions. The M6800 Microprocessor Applications Manual illustrates the PIA addressing and the operation being performed on the selected PIA register. Note that bit 2 of PIA control registers is used in selecting between the PIA's peripheral interface registers and the data direction registers. The proper state of the CA2 and CB2 bit must be loaded into the corresponding PIA control register before addressing its peripheral interface or data direction register.

The M6800 Microprocessor Applications Manual also discusses the decoding of the PIA control registers in controlling the PIA's four interrupt lines — two of which may be used as peripheral control lines. Refer to the Applications Manual and determine the contents to be loaded into the control registers to meet your application needs. Note that the MC6800 MPU cannot write into bit 6 and bit 7 of the control registers, but checks the status of the flag bits stored at these register locations during a read operation. Note also that the flag bits are reset when they are read.

The MC6820 PIA data direction registers determine whether their respective peripheral lines (PA0 through PA7 and PB0 through PB7) are to be used as input or output lines. A logic one in a data direction register bit position enables its corresponding peripheral line to function as an output and a logic zero causes this line to function as an input.

3-4.6 MC6850 ACIA PROGRAMMING INFORMATION

The MC6800 MPU addresses the MC6850 ACIA as if it were memory. Therefore, all commands to the ACIA are executed by the MPU as memory reference instructions. The M6800 Microprocessor Applications Manual illustrates the ACIA addressing and the operation being performed by the selected ACIA register. Note that the MPU can only read the receive data register and the status register and can only write into the transmit data register and control register.

The M6800 Microprocessor Applications Manual also discusses the function of each of the bits in

the control register and the function of the bits in the status register. Note that both flag bit 0 and bit 1 of the status register are reset when the status register is read.

3-4.7 PROGRAM PREPARATION

Prepare your program in accordance with the following procedures:

(a) Using flowcharts or other means design the user's program.

(b) Code the user's program as described in the M6800 Microprocessor Programming Manual

(c) Assemble the user's program using either the resident assembler or a cross-assembler.

NOTE

The user may write his program in machine language and enter his program through his data terminal using the MAID Memory Change Function in Paragraphs 4-9 and 4-11 of this manual. This is, however, a long and tedious job, subject to error, and therefore not recommended.

(d) Prepare the assembled user's program to be evaluated in the EXORciser.

3-5 SYSTEM DEBUG AND DEVELOPMENT

Now that you have set up the EXORciser and have assembled the user's program, you are ready to debug the user's system hardware and program. Debug the user's system as follows:

(a) Load the user's program into the EXORciser and, using the procedures in Chapter 4, debug the user's system hardware and software.

(b) Refer to the M6800 Microcomputer Applications Manual and design a preproduction model of the user's system. Refer to Chapter 5 of this manual and Chapter 3 of the supplements to get an overview of the user's system as emulated by the EXORciser. If you are going to use the EXORciser to debug the user's preproduction system, design this system's bus to be compatible with the EXORciser's bus except, connect the EXORciser's VUA (Valid User's Address) rather than its VMA (Valid Memory Address) signal line to VMA signal line on the user's system bus. The VUA signal is used to prevent duplicate addressing while the EXORciser is executing it EXbug Firmware.

(c) Debug your preproduction system. If you are using the EXORciser, remove the MPU chip from the user's system and debug the user's system using the basic EXORciser (just the MPU and Debug Modules). Debug the user's system using the procedures in Chapter 4.

(d) Design your production system. If you intend to use the EXORciser to debug this system to refer to step b for details.

(e) Debug your production system using the procedures discussed in step c.

CHAPTER 4
PROGRAM EVALUATION AND DEBUG PROCEDURES

4-1 GENERAL

The EXORciser with its EXbug firmware is used to evaluate and debug a program or a system under development. The EXbug firmware enables the user to:

- Load his program into the EXORciser (Paragraph 4-4)
- Verify that his program was properly loaded into the EXORciser (Paragraph 4-5)
- Search the tape for a particular file (Paragraph 4-6)
- Print the program (Paragraph 4-7)
- Punch (record) the program on tape (Paragraph 4-8)
- Using the MAID (Motorola's Active Interface Debug) routine, test and debug his program or system under development. (Paragraph 4-9)

The operating procedures for each of these functions as well as the abort function are discussed in the following paragraphs. The EXORciser, while performing its EXbug program, is inhibited from performing the user's program except under specific MAID control routines.

It is assumed in this chapter that the EXORciser has been installed in accordance with Chapter 2 and the EXORciser is interfacing with a data terminal described in Chapter 2. It is also assumed that the user's programs have been prepared in accordance with the procedures described in Chapter 3 and the EXORciser hardware has been configured to emulate the user's system as described in Chapter 3.

4-2 OPERATING PROCEDURES

NOTE

The location and use of the EXORciser indicators and controls are discussed and locations identified in Chapter 2.

(a) Set the EXBUG switch on the Debug Module to ON.

(b) Set the BAUD RATE switch to the selected transfer rate

(c) Turn the terminal power switch to ON.

(d) Using the power key, set the EXORciser POWER switch to ON. The EXORciser performs its EXbug control program and the terminal prints EXBUG x.x where x.x identifies the version of the EXbug program being used in the EXORciser. The EXORciser now waits for a four character command to be entered.

NOTE

The EXORciser is capable of operating with data terminals that record at one data transfer rate and print data at one fourth this data transfer rate by inserting the appropriate number of nulls between characters. If the EXORciser is interfacing with such a data terminal, enter S120X and pause before entering the X.

(e) If the first few characters are missing from the displayed data and all of the other characters are correct, there is insufficient delay between the carriage return and the first character to be printed. Enter S30. and the EXbug program will insert an appropriate number of nulls. (4)

(f) If you are not using the address 83FF as the top memory address in your user's program, enter the top memory of your program address at memory location FF00 and FF01 using the MAID Examine and Change Memory Data Function in accordance with Paragraphs 4-10 and 4-11.

4-3 ABORT FUNCTION

If the EXORciser in debugging a user's program should lose control or you wish to exit from an EXORciser function, perform the following procedures.

(a) Press the ABORT switch. The EXORciser should return to the EXbug control program and the data terminal should print the following message:

ABORTED AT P-0106 X-4000 A-04 B-00 C-C4 S-FF91
EXBUG x.x

where:
- P = program counter
- X = index register
- A = accumulator A
- B = accumulator B
- C = condition code register
- S = stack pointer

NOTE

The values in the counters and registers are typical.

(b) If the pressing of the ABORT switch does not regain program control, press the RESTART switch. This causes the EXORciser to initiate a power-on interrupt routine and initialize itself. Reenter the data terminal's baud rate and the top address of the user's program (if not 83FF) in accordance with Paragraph 4-2.

4-4 MEMORY LOADER FUNCTION (LOAD)

The Memory Loader Function of EXbug loads formatted binary object tapes into the EXORciser. Figure 4-1 depicts the paper tape format. It is assumed at the start of this procedure that the EXORciser is performing its EXbug control program and the last data printed by the terminal is EXBUG x.x. Figure 4-2 illustrates a typical Memory Loader Function.

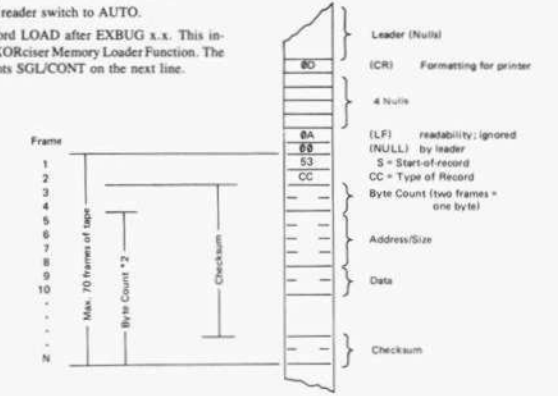
(a) Load the tape into the terminal tape reader.

(b) Set the tape reader switch to AUTO.

(c) Enter the word LOAD after EXBUG x.x. This initiates the EXORciser Memory Loader Function. The terminal prints SGL/CONT on the next line.

NOTE

To abort from the Memory Loader Function, enter the character X. The Memory Loader Function also may be aborted by pressing the ABORT switch. Should you make a typing error in entering the word LOAD, type in a sufficient number of characters to form a four character code. The EXORciser will remain in its EXbug control program and the terminal will print EXBUG x.x on the next line. Repeat step c.



Frames 3 through N are hexadecimal digits (in 7-bit ASCII) which are converted to BCD. Two BCD digits are combined to make one 8-bit byte.

The checksum is the one's complement of the summation of 8-bit bytes.

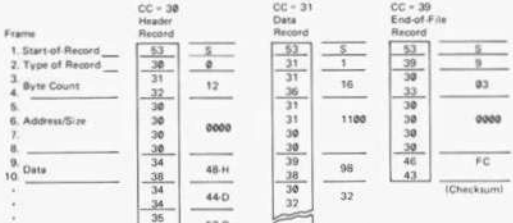


FIGURE 4-1. Paper Tape Format

(d) Enter the character S or the character C after SGL/CONT. The character S initiates the loading of one file into the EXORciser and then returns to the EXbug control program. The character C, on the other hand, initiates the loading of multiple files into the EXORciser. If the character S is entered, proceed to the following Checksum Error Detection paragraph below.

NOTE

If the EXORciser attempts to load the data into ROM, protected RAM, or non-existent memory locations, the tape reader stops, the terminal prints NO CHANGE, and the EXORciser returns to the EXbug control program.

(e) If the character C was entered in step d, press the ABORT switch at the conclusion of the memory loader function. The EXORciser now returns to the EXbug control program and the terminal prints EXBUG x.x.

CHECKSUM ERROR DETECTION

If in the loading function, the EXORciser detects a checksum error, it instructs the teletypewriter to print CKSM nnnn and then it stops the tape reader. The nnnn in the checksum error is the address of the record found in error.

(f) If a checksum error message is present, perform one of the following substeps:

(X) Enter the character X and abort from the Memory Loader Function. The EXORciser will return to the EXbug control program and the terminal will print EXBUG x.x.

(L) Reposition the tape and enter the character R. The record causing the checksum error is reread.

(C) Ignore the checksum error and enter the character C. The EXORciser ignores the checksum error and continues the Memory Loader Function.

CAUTION

IF THE CHECKSUM ERROR IS ON AN ADDRESS AND THE CONTINUE OPTION IS SELECTED, THERE IS NO CERTAIN WAY TO DETERMINE WHERE THE DATA WILL BE LOADED INTO THE MEMORY.

NOTE

The underlined characters in the figures depicts operate entries.

```
EXBUG 1.1 LOAD
SGL/CONT S
 X EXBUG11
EXBUG 1.1
```

FIGURE 4-2. Typical Memory Loader Function

4-5 MEMORY VERIFY FUNCTION (VERF)

The Memory Verify Function of EXbug compares the contents of memory with the paper tape data and prints all locations that are not the same. Figure 4-1 depicts the paper tape format. It is assumed that the EXORciser is performing its EXbug control program and the last data printed by the terminal is EXBUG x.x. Figure 4-3 illustrates a typical Memory Verify Function.

(a) Load the tape into the terminal tape reader.

(b) Set the tape reader switch to AUTO.

(c) Enter the word VERF after EXBUG X.X. This initiates the EXORciser Memory Verify Function. The terminal shall print SGL/CONT on the next line.

NOTE

To abort from the Memory Verify Function enter the character X. The Memory Verify Function also may be aborted by pressing the ABORT switch. Should you make a typing error in entering the word VERF, type in a sufficient number of characters to form a four character code. The EXORciser will remain in its EXbug control program and the terminal will print EXBUG x.x on the next line. Repeat step c.

(d) Enter the character S or the character C after SGL/CONT. The character S initiates the verification of one memory file and stops at the end of the file. The character C, on the other hand, initiates the verification of multiple files in the EXORciser. If the character S was entered in step d, proceed to the checksum error detection paragraph.

(e) If the character C was entered in step d, press the ABORT switch at the conclusion of the memory verify function. The EXORciser returns to the EXbug control program and the terminal prints EXBUG x.x.

```
EXBUG 1.1 VERF
SGL/CONT S
HDR
ADDR/MEM/TAPE
0013 80 DE
001C 12 06
EXBUG 1.1
```

FIGURE 4-3. Typical Memory Verify Function

CHECKSUM ERROR DETECTION

If, in the verification, the EXORciser detects a checksum error, it instructs the terminal to print CKSM nnnn and then stops the tape reader. The nnnn in the checksum error is the address of the record found in error.

(f) If a checksum error message is printed, perform one of the following substeps:

(X) Enter the character X and abort the Memory Verify Function. The EXORciser returns to the

EXbug control program and the terminal prints EXBUG x.x.

(R) Reposition the tape and enter the character R. The record causing the checksum error is reread.

(C) Ignore checksum error and enter the character C. The EXORciser ignores the checksum error and continues the Memory Verify Function.

NOTE

If the checksum error is on an address and the continue option is selected, the verification process may be meaningless.

4-6 TAPE SEARCH FUNCTION (SRCH)

The EXORciser in the Tape Search Function searches the paper tape until it encounters a header record. The EXORciser, on detecting the header record, stops the tape reader and prints the header record. The operator now has the option of continuing the search, switching to the Memory Loader Function, or switching to the Verify Memory Function. Figure 4-4 illustrates a typical Tape Search Function.

```
EXBUG 1.1 SRCH
HDR
CONT/LOAD/VERF C
X EXORTS2
CONT/LOAD/VERF L
SGL/CONT S
EXBUG 1.1 VERF
SGL/CONT S
X EXORTS2
ADDR/MEM/TAPE
0019 8D 08
001C 12 08
002F DE 02
EXBUG 1.1
```

FIGURE 4-4. Typical Tape Search Function

(a) Load the tape into the terminal tape reader.

(b) Set the tape reader switch to AUTO.

(c) Enter the word SRCH after EXBUG x.x. This initiates the Tape Search Function. The EXORciser advances tape until it detects a header record and then stops the tape reader. The terminal now prints the text from the header and on the next line prints CONT-/LOAD/VERF. The operator now has the option of: (C) continuing the tape search, (L) loading the file into memory, (V) verifying the file, or (X) aborting the tape search program and returning to the EXbug control program.

NOTE

Should you make a typing error in entering the word SRCH, type in a sufficient number of characters to form a four character code. The EXORciser will remain in its EXbug control program and the data terminal will print EXBUG X.X on the next line. Repeat step c.

(d) Refer to Table 4-1 and enter the appropriate character after CONT/LOAD/VERF to select the desired operation.

4-7 PUNCH MEMORY DUMP FUNCTION

The Punch Memory Dump Function instructs the EXORciser to punch an absolute formatted binary object tape. Figure 4-1 depicts the paper tape format. It is assumed that the EXORciser is performing its EXbug control program and the last data printed by the terminal is EXBUG x.x. Figure 4-5 illustrates a typical Punch Memory Dump Function.

NOTE

To abort from the Punch Memory Dump Function, enter the character X. The Punch Memory Dump Function also may be aborted by pressing the ABORT switch.

(a) Prepare the tape reader to punch a tape.

(b) Enter the word PNCH after EXBUG x.x. This initiates the EXORciser Punch Memory Dump Function. The terminal prints BEG ADDR nnnn on the next line.

NOTE

The nnnn in the BEG ADDR and END ADDR is the address used in the previous Punch Memory Dump Function. Also, should you make a typing error in entering the word PNCH, type in a sufficient number of characters to form a four character code. The EXORciser will remain in its EXbug control program and the terminal will print EXBUG x.x on the next line. Repeat step b.

TABLE 4-1. Tape Search Program Character Selection

CHARACTER	FUNCTION
C	Continues searching the tape.
L	Loads the file into memory. The EXORciser automatically transfers to the memory load program and prints SGL/CONT. Proceed from Paragraph 4-4 step d.
V	Verifies the file. The EXORciser automatically transfers to the memory verify program and prints SGL/CONT. Proceed from Paragraph 4-5 step d.
X	Aborts the tape search function. The EXORciser prints EXbug x.x and returns to the EXbug control program.

```
EXBUG 1.1 PNCH
BEG ADDR 024A 10
END ADDR 0D9B 38
HDR=X EXORT2
EXEC Y
S00B0000582045584F525432B9
S11300109E005996009709860F9709DE0BDFA9B7C
S1130020A700A1002606093C000B26F17EF56402C8
S10F003000018000E0003121622223202BDF
S9030000FC
BEG ADDR 0010 X
EXBUG 1.1
```

FIGURE 4-5. Typical Punch Memory Dump Function

(c) Enter in hexadecimal the selected beginning address after BEG ADDR nnnn and then enter a CR (Carriage Return) character. The terminal prints END ADDR nnnn on the next line.

NOTE

Should you make a typing error in entering the hexadecimal beginning or ending address, type in a sufficient number of hexadecimal digits to form a five character code. Do not enter a carriage return as the fifth character. The EXORciser will disregard this address and the terminal will print BEG ADDR nnnn or END ADDR nnnn on the next line.

(d) Enter in hexadecimal the selected end address after END ADDR nnnn and then enter a CR (Carriage Return) character. The terminal prints HDR = X on the next line.

NOTE

If the END ADDR is smaller than the BEG ADDR, the EXORciser will disregard these addresses and the terminal will print BEG ADDR nnnn. Repeat steps c and d.

(e) Enter the six-character header record name after the HDR =X. The terminal prints EXEC on the next line.

(f) Enter the character Y after EXEC. The EXORciser initiates the punching operation. At the conclusion of the punch operation the terminal prints BEG ADDR nnnn.

NOTE

The nnnn in the BEG ADDR nnnn now is the address entered in step c.

The user now must decide whether to punch another tape of this file, to punch another file, or to exit from this function.

(g) At the conclusion of punching the tape, perform one of the following substeps:

(1) To punch a second tape of the file, repeat steps b through e, however, enter only a CR (Carriage Return) character after the address in steps c and d.

(2) To punch another file on tape, repeat steps b through e and enter the beginning and concluding addresses of the new file in steps c and d.

(3) To exit from the punch memory dump function, enter a character X. The EXORciser will return to the EXbug control program and the terminal prints EXBUG x.x.

4-8 PRINT MEMORY DUMP FUNCTION (PRNT)

The Print Memory Dump Function instructs the terminal to print the contents of memory in a hexadecimal format followed by the literal ASC II characters. Figure 4-6 illustrates a typical Print Memory Dump Function. It is assumed that the EXORciser is performing its EXbug control program and the last data printed by the terminal is EXBUG x.x.

```
EXBUG 1.1 PRNT
BEG ADDR 00A0 FBA0
END ADDR FBB6
EXEC Y
FBA0 53 54 4F 50 20 41 44 44 52 20 04 41 42 4F 52 54  STOP ADDR  .ABORT
FBB0 45 44 20 41 54 20 04 14 13 00 00 00 00 04 07 20  ED AT  ..........
BEG ADDR FBA0 X
EXBUG 1.1
```

FIGURE 4-6. Typical Print Memory Dump Function

NOTE

To abort from the Print Memory Dump Function, enter the character X. The Print Memory Dump Function also may be aborted by pressing the ABORT switch.

(a) Enter the word PRNT after EXBUG x.x. This initiates the EXORciser Print Memory Dump Function. The terminal prints BEG ADDR nnnn on the next line.

NOTE

The nnnn in the BEG ADDR and END ADDR is the address used in the previous Print Memory Dump Function. Also, should you make a typing error in entering the word PRNT, type in a sufficient number of characters to form a four character code. The EXORciser will remain in its EXbug control program and the terminal will print EXBUG x.x on the next line. Repeat step a.

(b) Enter in hexadecimal the selected beginning address after BEG ADDR nnnn and then enter a CR (Carriage Return) character. The terminal prints END ADDR nnnn on the next line.

NOTE

Should you make a typing error in entering a hexadecimal beginning or ending address, type in a sufficient number of hexadecimal digits to form a five character code. Do not enter a carriage return as the fifth character. The EXORciser will disregard this address and the terminal will print BEG ADDR nnnn or END ADDR nnnn on the next line.

(c) Enter in hexadecimal the selected end address after END ADDR nnnn and then enter a CR character. The terminal prints EXEC on the next line.

NOTE

If the END ADDR is smaller than the BEG ADDR, the EXORciser will disregard these addresses and the terminal will print BEG ADDR nnnn. Repeat steps b and c.

(d) Enter the character Y after EXEC. The EXORciser initiates the memory printing operation. At the conclusion of the printing operation the terminal prints BEG ADDR nnnn.

NOTE

The nnnn in the BEG ADDR now is the address entered in step c.

The user now must decide whether to reprint the selected file, to print another file, or to exit from this function.

(e) At the conclusion of printing the file perform one of the following substeps:

(1) To print a second copy of the file, repeat steps a through e; however, enter only a CR (Carriage Return) character after the address in steps a and b.

(2) To print another file, repeat steps a through c and enter the beginning and end addresses of the new file in steps b and c.

(3) To exit from the print memory dump function, enter a character X. The EXORciser returns to the EXbug control program and the terminal prints EXBUG x.x on the next line.

4-9 MAID FUNCTION PROCEDURES

The MAID (Motorola Aided Interface Debug) function of the EXORciser EXbug control program enables the user to perform the following functions in debugging his progam.

- Examine and change data in a memory location. (Paragraph 4-11)
- Calculate the offset in the relative addressing mode. (Paragraph 4-12)
- Examine and change the data in the MPU program registers and counters. (Paragraph 4-13)
- Insert, display, and remove breakpoints in the program. (Paragraph 4-14)
- Stop the EXORciser on a selected memory address (Paragraph 4-15)
- Provide a scope trigger pulse on a selected memory address (Paragraph 4-16)
- Freerun or trace through the user's program under MAID control. (Paragraph 4-17 through 4-19)
- Perform decimal-octal-hexadecimal conversions. (Paragraph 4-21)
- Search memory for a bit pattern. (Paragraph 4-20)

Table 4-2 identifies each of the MAID Commands used in debugging the user's program. The following paragraphs discuss placing the EXORciser in the MAID function and performing each of the MAID function operations. It should be recalled that the EXORciser in EXbug is inhibited from performing the user's program except under specific MAID functions.

4-10 ENTERING MAID FUNCTION

It is assumed at the start of this function that the EXORciser is performing its EXbug control program and the last data printed by the terminal is EXBUG x.x.

Enter the word MAID after EXBUG x.x. This places the EXORciser in the MAID function. The terminal prints *(an asterisk) on the next line.

TABLE 4-2. MAID Program Control Commands

MAID COMMAND	DESCRIPTION
n/	Print the contents of memory location n and enable the EXORciser to change the contents of this memory location.
n;O	Calculate the address offset (for relative addressing mode instructions).
(LF)	Print the contents of the next sequential memory location and enable the EXORciser to change the contents of this memory location. (LF — Line Feed character)
↑	Print the contents of the previous sequential memory location and enable the EXORciser to change the contents of this memory location. (↑ — up arrow character or SHIFT key and N character).
(CR)	Return the displayed contents to memory and accept next command (CR — Carriage Return character).
n;V	Enter a breakpoint at memory location n.
$V	Display the memory location of each breakpoint
n;P	Continue executing from the selected breakpoint until this breakpoint is encountered n times.
n;U	Remove the breakpoint at memory location n.
;U	Remove all the breakpoint.
n;W	Search for the n bit pattern.
$M	Display the search mask.
;G	Execute the user's program starting at the auto restart memory location.
n;G	Execute user's program starting at memory location n.
$R	Display/change the user's program registers.
;P	Continue executing from the current program counter setting.
;N	Trace one instruction.
N	Trace one instruction.
n;N	Trace n instructions.
$T	Set the trace mode.
;T	Reset the trace mode.
$S	Display and set the stop-on-address compare — Scope trigger pulse.
;S	Reset the stop-on-address compare — Scope trigger pulse.
#n =	Convert the decimal number n to its hexadecimal equivalent.
#$n =	Convert the hexadecimal number n to its decimal equivalent.
#@n =	Convert the octal number n to its hexadecimal equivalent.

4-11 EXAMINE AND CHANGE MEMORY DATA FUNCTION

The EXORciser performs this function in three steps: 1) examining the contents of a selected memory location (opening the memory location), 2) changing the contents of this location if required, and 3) closing the memory location. The EXORciser, in examining a memory location, instructs the terminal to print the contents of this memory location.

The EXORciser in this function displays each of the program instructions in machine language. Table 4-3 is an M6800 instruction map to assist the user in reading and changing his program steps.

It is assumed at the start of this procedure that the EXORciser is performing its MAID control program and the last data printed by the terminal is an asterisk.

Figure 4-7 depicts a Typical Memory Change Function.

```
EXBUG 1.1 MAID
*800/00 12

NO CHANGE
*400/01 24
0401/89
0402/05 23
0403/1F
0402/23 66
0403/1F
*
```

FIGURE 4-7. Typical Examine and Change Memory Data Operation

LSB / MSB	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	*	NOP (INH)	*	*	*	*	TAP (INH)	TPA (INH)	INX (INH)	DEX (INH)	CLV (INH)	SEV (INH)	CLC (INH)	SEC (INH)	CLI (INH)	SEI (INH)
1	SBA	CBA	*	*	*	*	TAB (INH)	TBA (INH)	*	DAA (INH)	*	ABA (INH)	*	*	*	*
2	BRA (REL)	*	BHI (REL)	BLS (REL)	BCC (REL)	BCS (REL)	BNE (REL)	BEQ (REL)	BVC (REL)	BVS (REL)	BPL (REL)	BMI (REL)	BGE (REL)	BLT (REL)	BGT (REL)	BLE (REL)
3	TSX (INH)	INS (INH)	PUL (A)	PUL (B)	DES (INH)	TXS (INH)	PSH (A)	PSH (B)	*	RTS (INH)	*	RTI (INH)	*	*	WAI (INH)	SWI (INH)
4	NEG (A)	*	*	COM (A)	LSR (A)	*	ROR (A)	ASR (A)	ASL (A)	ROL (A)	DEC (A)	*	INC (A)	TST (A)	*	CLR (A)
5	NEG (B)	*	*	COM (B)	LSR (B)	*	ROR (B)	ASR (B)	ASL (B)	ROL (B)	DEC (B)	*	INC (B)	TST (B)	*	CLR (B)
6	NEG (IND)	*	*	COM (IND)	LSR (IND)	*	ROR (IND)	ASR (IND)	ASL (IND)	ROL (IND)	DEC (IND)	*	INC (IND)	TST (IND)	JMP (IND)	CLR (IND)
7	NEG (EXT)	*	*	COM (EXT)	LSR (EXT)	*	ROR (EXT)	ASR (EXT)	ASL (EXT)	ROL (EXT)	DEC (EXT)	*	INC (EXT)	TST (EXT)	JMP (EXT)	CLR (EXT)
8	SUB (A) (IMM)	CMP (A) (IMM)	SBC (A) (IMM)	*	AND (A) (IMM)	BIT (A) (IMM)	LDA (A) (IMM)	*	EOR (A) (IMM)	ADC (A) (IMM)	ORA (A) (IMM)	ADD (A) (IMM)	CPX (A) (IMM)	BSR (REL)	LDS (IMM)	*
9	SUB (A) (DIR)	CMP (A) (DIR)	SBC (A) (DIR)	*	AND (A) (DIR)	BIT (A) (DIR)	LDA (A) (DIR)	STA (A) (DIR)	EOR (A) (DIR)	ADC (A) (DIR)	ORA (A) (DIR)	ADD (A) (DIR)	CPX (A) (DIR)	*	LDS (DIR)	STS (DIR)
A	SUB (A) (IND)	CMP (A) (IND)	SBC (A) (IND)	*	AND (A) (IND)	BIT (A) (IND)	LDA (A) (IND)	STA (A) (IND)	EOR (A) (IND)	ADC (A) (IND)	ORA (A) (IND)	ADD (A) (IND)	CPX (A) (IND)	JSR (IND)	LDS (IND)	STS (IND)
B	SUB (A) (EXT)	CMP (A) (EXT)	SBC (A) (EXT)	*	AND (A) (EXT)	BIT (A) (EXT)	LDA (A) (EXT)	STA (A) (EXT)	EOR (A) (EXT)	ADC (A) (EXT)	ORA (A) (EXT)	ADD (A) (EXT)	CPX (A) (EXT)	JSR (EXT)	LDS (EXT)	STS (EXT)
C	SUB (B) (IMM)	CMP (B) (IMM)	SBC (B) (IMM)	*	AND (B) (IMM)	BIT (B) (IMM)	LDA (B) (IMM)	*	EOR (B) (IMM)	ADC (B) (IMM)	ORA (B) (IMM)	ADD (B) (IMM)	*	*	LDX (IMM)	*
D	SUB (B) (DIR)	CMP (B) (DIR)	SBC (B) (DIR)	*	AND (B) (DIR)	BIT (B) (DIR)	LDA (B) (DIR)	STA (B) (DIR)	EOR (B) (DIR)	ADC (B) (DIR)	ORA (B) (DIR)	ADD (B) (DIR)	*	*	LDX (B) (DIR)	STX (B) (DIR)
E	SUB (B) (IND)	CMP (B) (IND)	SBC (B) (IND)	*	AND (B) (IND)	BIT (B) (IND)	LDA (B) (IND)	STA (B) (IND)	EOR (B) (IND)	ADC (B) (IND)	ORA (B) (IND)	ADD (B) (IND)	*	*	LDX (IND)	STX (IND)
F	SUB (B) (EXT)	CMP (B) (EXT)	SBC (B) (EXT)	*	AND (B) (EXT)	BIT (B) (EXT)	LDA (B) (EXT)	STA (B) (EXT)	EOR (B) (EXT)	ADC (B) (EXT)	ORA (B) (EXT)	ADD (B) (EXT)	*	*	LDX (EXT)	STX (EXT)

DIR = Direct Addressing Mode
EXT = Extended Addressing Mode
IMM = Immediate Addressing Mode
IND = Index Addressing Mode
INH = Inherent Addressing Mode
REL = Relative Addressing Mode

A = Accumulator A
B = Accumulator B

TABLE 4-3. M6800 Instruction Map

NOTE

If the memory address selected is in ROM or protected RAM, the contents of this memory location cannot be changed.

(a) Enter the address of the memory location to be examined and the a / (slash character). The EXORciser examines this memory location and the terminal prints the contents of this memory location in hexadecimal.

NOTE

If the memory contents are not to be changed, proceed to step c.

(b) If the contents of the memory location are to be changed, enter in hexadecimal the new data to be stored at this location.

The operator now must decide in closing this memory location whether to return to MAID control program, to examine the previous sequential memory location, or to examine the following sequential memory location.

(c) Perform one of the following substeps and close this memory location.

(1) Enter a CR (Carriage Return) character. The EXORciser closes the memory location and returns to the MAID function. The terminal prints an asterisk character on the next line.

(2) Enter an ↑ (up arrow) character (or SHIFT key and N character). The EXORciser closes this memory location and examines the previous sequential memory location. The terminal now prints the new memory address, a / character and the contents of this memory location. Return to step b.

(3) Enter a LF (Line Feed) character. The EXORciser closes this memory location and examines the next sequential memory location. The terminal now prints the new memory address, a / character and the contents of this memory location. Return to step b.

NOTE

If the memory address selected to be changed is in ROM or protected RAM, the data can not be changed and the terminal prints NO CHANGE on the next line, and the EXORciser returns to the MAID control program.

4-12 CALCULATE RELATIVE MODE OFFSET FUNCTION

The EXORciser in this function calculates whether the branch address in the second byte of a relative mode instruction is within a range of −127 to +128 bytes from the next location. It is assumed at the start of this procedure that the EXORciser is performing its MAID control program and the last data printed by the terminal is as asterisk. Figure 4-8 depicts a Typical Relative Mode Offset Function.

```
X
EXBUG 1.1 MAID
*0400/01 4813;O INVL
0400/01 4803;O 7F
0400/01 7F
0401/89
*0480/0B 4003;O INVL
0480/0B 4013;O 80
0480/0B 80
0481/09
0482/05 ↑
0481/09 7
0482/05
*
```

FIGURE 4-8. Calculate Relative Mode Offset Operation

(a) Enter the address of the memory location who's offset is to be calculated and then enter a / (slash) character. The EXORciser opens this memory location and the terminal prints the contents of this memory location in hexadecimal.

(b) Enter the relative address you wish to branch to followed by a ;O (semi-colon and the letter O). The EXORciser calculates the offset to this address. The terminal prints the offset if it is within the −127 to +128 byte range or INVL (INVaLid) if it is not within range. In either case the terminal reprints the memory address, a / character, and the memory contents on the next line.

(c) If the offset address entered in step b is within range, enter the offset calculated in step b followed by a CR character. If, however, the offset address entered in step b is INVL you will have to modify your program.

4-13 EXAMINE AND CHANGE MPU PROGRAM REGISTER FUNCTION

The EXORciser in each of the following procedures, performs a simulated powerfail routine and returns to the MAID function.

Execute-one-instruction occurs.

Breakpoints are reached.

Stop-on-address is recognized.

The EXORciser at this time stores the contents of the MPU register in memory (pseudo stack), prints the data stored on the pseudo stack, and returns from the MAID masked user's program to the MAID control program. The user, through the Examine and Change MPU Program Register Function, has the capability of examining and changing the contents of the MPU registers stored on the pseudo stack. It is assumed at the start of this operation that the EXORciser is performing its MAID function and the last data printed by the terminal is an asterisk. Figure 4-9 depicts a Typical Examine and Change MPU Register Function.

```
EXBUG 1.1 MAID
*$R
P-0106 X-4000 A-04 B-00 C-C4 S-FF91
P-0105 100
X-4000 2345
A-04
B-00 44
*
```

FIGURE 4-9. Typical Examine and Change User Program Register Function

(a) Enter the characters $R. The EXORciser now is enabled to perform the Examine and Change MPU Program Register Function and the terminal prints:
P-0106 X-4000 A-04 B-00 C-C4 S-FF91

where: P = program counter
X = index register
A = accumulator A
B = accumulator B
C = condition code register
S = stack pointer

NOTE
The values in the registers are typical. Table 4-4 depicts the status of the condition code register.

(b) If none of the registers require change, enter a CR (Carriage Return) character. The EXORciser returns to the MAID function and the terminal prints an asterisk on the next line. If, however, one of the registers requires change, proceed to step c.

(c) If the contents of one of the user program registers requires change, using Table 4-5, change the data in the selected MPU program register.

NOTE
Only a line feed or a carriage return character can be used in this function. The only way to go back to a register is to enter a carriage return character and repeat this function.

TABLE 4-4. Condition Code Register Status

CONDITION CODE REGISTER BITS								STATUS BIT BEING MONITORED
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	
1	1	X	X	X	X	X	0	carry/borrow clear
1	1	X	X	X	X	X	1	carry/borrow set
1	1	X	X	X	X	0	X	overflow clear
1	1	X	X	X	X	1	X	overflow set
1	1	X	X	X	0	X	X	zero clear
1	1	X	X	X	1	X	X	zero set
1	1	X	X	0	X	X	X	negative clear
1	1	X	X	1	X	X	X	negative set
1	1	X	0	X	X	X	X	interrupt mask clear
1	1	X	1	X	X	X	X	interrupt mask set
1	1	0	X	X	X	X	X	half carry clear
1	1	1	X	X	X	X	X	half carry set

TABLE 4-5. Changing User Program Data

REGISTER	REQUIRE CHANGE	
	YES	NO
Program Counter	Enter new data and a LF character	Enter a LF character
Index Register	Enter new data and a LF character	Enter a LF character
Accumulator A	Enter new data and a LF character	Enter a LF character
Accumulator B	Enter new data and a LF character	Enter a LF character
Condition Code Register	Enter new data and a LF character	Enter a LF character
Stack Pointer	Enter a new data and a CR character	Enter a CR character

4-14 INSERT; DISPLAY; AND REMOVE BREAKPOINT FUNCTION

This function enables the EXORciser to insert up to eight breakpoints at specific points in the user program, to display the breakpoint addresses on command, and to remove the breakpoints. During a MAID execute function (Paragraph 2) the breakpoints are inserted into the user's program and instruct the EXORciser to stop at the selected breakpoint addresses. The breakpoint sequence is:

- User designates the breakpoint(s) address(es).
- User initiates the execute function.
- The EXORciser replaces the program instruction(s) with breakpoint(s) and transfers control to the user's program.
- The EXORciser on detecting a breakpoint instruction returns to the MAID program control routine.
- The MAID function inserts the program instruction and the terminal prints the contents of the MPU program registers.

It is assumed at the start of this function that the EXORciser is performing its MAID function and the last data printed by the terminal is an asterisk. Figure 4-10 depicts 1) the insertion of breakpoints, 2) the displaying of breakpoints, and 3) the resetting of breakpoints.

Entering a breakpoint (;V):

(a) Enter the address of the selected breakpoint.

NOTE

Memory address 0000 is used to indicate no breakpoints. Therefore, no breakpoints can be inserted at address 0000.

(b) After the breakpoint address, enter a;V (semi-colon and the character V). The EXORciser enters the breakpoint in the user's program and returns to its MAID function. The terminal prints an asterisk on the next line.

Displaying a breakpoint ($V):

(c) Enter the characters $V. The EXORciser displays the breakpoints and then returns to the MAID function. The terminal prints the memory address of each breakpoint.

NOTE

If no breakpoints have been entered, the terminal prints the address 0000.

Remove (reset) breakpoints (n;U or ;U)

The user has the option of removing one breakpoint or all breakpoints at one time. To remove all breakpoints, enter ; U (semicolon and then character U).

(d) Enter the memory address of the breakpoint to be removed.

(e) After the selected breakpoint address, enter ; (semi-colon) and the character U. The EXORciser removes the breakpoint from the user program and returns to the MAID function. The terminal prints an asterisk on the next line.

```
EXBUG 1.1 MAID
*20;V
*10;G
A
P-0020 X-000F A-41 B-F0 C-C0 S-0050
*2A;V
*20;U
*;P
P-002A X-000E A-41 B-F0 C-C0 S-0050
*3;PASD
P-002A X-000B A-44 B-F0 C-C4 S-0050
*$V
0000 002A 0000 0000 0000 0000 0000 0000
*;U
*$V
0000 0000 0000 0000 0000 0000 0000 0000
*
```

FIGURE 4-10. Insert; Display; and Remove Breakpoint Functions

4-15 STOP-ON-ADDRESS FUNCTION

The EXORciser, on reaching the address entered in this function, stops the user program on this address and returns to the MAID control program. It is assumed at the start of this function that the EXORciser is performing the MAID function and the last data printed by the terminal is an asterisk. Figure 4-11 depicts a typical Stop-On-Address Function.

```
EXBUG 1.1 MAID
*$S
STOP ADDR 0114 109
*101;G
STOP-ON-ADDRESS P-010C X-FF8D A-00 B-FF C-D4 S-FF8A
*;S
*
```

FIGURE 4-11. Typical Stop-On-Address Function

Set Stop-On-Address ($S):

(a) Set the SCOPE/STOP-ON-ADDRESS switch on the Debug Module to STOP-ON-ADDRESS.

(b) Enter the characters $S. The EXORciser sets the Stop-On-Address Function and the terminal prints STOP ADDR nnnn 0000 on the next line where nnnn is the address being monitored.

(c) Enter the stop address and then a CR (Carriage Return) character. The EXORciser inserts the Stop-On-Address in the user program and returns to the MAID function. The terminal prints an asterisk on the next line.

NOTE

The EXORciser in executing the user's program with the stop-on-address routine enabled, monitors the EXORciser bus. On reaching the selected address in the users program the EXORciser, completes the instruction it is performing and then goes to the MAID control program. Since the MPU instructions are one, two, and three bytes long, the EXORciser may not stop on the selected address but stops on the address of the next instruction.

Reset Stop-On-Address (;S)

(d) Enter the characters; (semi-colon) S. The EXORciser disables the Stop-On-Address Function and returns to the MAID function. The terminal prints an asterisk on the next line.

4-16 SCOPE TRIGGER FUNCTION

The Scope Trigger Function enables the EXORciser to generate a scope trigger pulse on the selected memory address. It is assumed at the start of this function that the EXORciser is performing the MAID function and the last data printed by the terminal is an asterisk. Figure 4-12 depicts a typical Scope Trigger Function.

```
EXBUG 1.1 MAID
*$S
STOP ADDR 0041 189
```

FIGURE 4-12. Typical Scope Trigger Function

Entering the Selected Scope Address ($S)

(a) Set the SCOPE/STOP-ON-ADDRESS switch on the Debug Module to SCOPE.

(b) Enter the characters $S. The EXORciser sets the Stop-On-Address Function and the terminal prints STOP ADDR nnnn on the next line where nnnn is the address being monitored.

(c) Enter the stop address and then a CR (Carriage Return) character. The EXORciser inserts the selected scope address in the program and returns to the MAID function. The terminal prints an asterisk on the next line.

Removing the Selected Scope Address

(d) Enter the characters ; (semi-colon) S. The EXORciser disables the selected scope address and returns to the MAID function. The terminal prints an asterisk on the next line.

4-17 EXECUTE USER'S PROGRAM FUNCTION

The user has the option of executing his program in either the Freerunning User's Program Function or the Trace User's Program Function. The differences between these two functions is that in the Trace User's Program Function the EXORciser prints the contents of

the program registers after each program instruction. The Freerunning User's Program Function procedures are presented in Paragraph 4-18 and the Trace User's Program Function procedures are presented in Paragraph 4-19.

4-18 FREERUNNING USER'S PROGRAM FUNCTION

This function enables the EXORciser to perform the user's program or a portion of the user's program in real time. It is assumed at the start of this function that the EXORciser is performing its MAID function and the last data printed by the terminal is an asterisk. Figure 4-13 depicts a typical Freerunning User's Program Function.

```
EXBUG 1.1 MAID
*10;G
EXBUG 1.1
```

FIGURE 4-13. Freerunning User Program Function

Refer to Table 4-6 and enter the selected command. The EXORciser shall perform the user's program until it reaches one of the following conditions:

- Detects a breakpoint (when the command n;P is entered the EXORciser on detecting a breakpoint decrement the count n by one and when n equals zero, exits the user's program.)
- Detects an interrupt wait (WAI) instruction. The program will restart on a non-masked I/O interrupt.
- Detects a stop-on-address compare.
- User's program loses control (recover control by pressing the ABORT or RESTART switch.)

4-19 TRACE USER'S PROGRAM FUNCTION

In the Trace User's Program Function the EXORciser prints the contents of the program register after the execution of each program step. There are two variations of the Trace User's Program Function — trace n instructions and trace to ending address. It is assumed at the start of this function that the EXORciser is performing the MAID function and the last data printed by the terminal is an asterisk.

Figure 4-14 depicts (1) trace n instructions and (2) trace to ending address.

```
EXBUG 1.1 MAID
*;R
P-001D X-000F A-0F B-00 C-D0 S-0050
P-001D 10
X-000F 0
*;N
P-0013 X-0000 A-0F B-00 C-D0 S-0050
P-0015 X-0000 A-00 B-00 C-D4 S-0050
P-0016 X-0000 A-00 B-00 C-D4 S-0050
*;T
END ADDR 001D 17
*10;G
P-0013 X-0000 A-00 B-00 C-D0 S-0050
P-0015 X-0000 A-00 B-00 C-D4 S-0050
P-0017 X-0000 A-00 B-00 C-D4 S-0050
*;T
*
```

FIGURE 4-14. Trace User Program Function

TABLE 4-6. Freerunning User Program Function Commands

COMMAND	OPERATION
;G	Initiate a restart interrupt and go.
n;G	Set the program counter to address n and go.
;P	Go from present address in the program counter.
n;P	Select number of times n EXORciser is to bypass the breakpoint and go.

4-19.1 TRACE N INSTRUCTIONS

(a) To trace one instruction, enter the character N or ; (semi-colon) N. The EXORciser traces one instruction and then returns to the MAID function. The terminal prints:

P-1003 X-0000 A-00 B-00 C-CO S-1100

where:
- P = program counter
- X = index register
- A = accumulator A
- B = accumulator B
- C = condition code register
- S = stack pointer

NOTE

The values in the registers are typical.

(b) To trace a selected number of instructions, enter the selected number of instructions to be traced (n), a semi-colon, and the character N. The EXORciser will trace the selected number of instructions and then return the MAID control program. The terminal prints the contents of the program registers for each instruction and an asterisk after the last instructions.

4-19.2 TRACE TO ENDING ADDRESS ($T)

(a) Enter the $T. The EXORciser shall initiate the Trace to Ending Address Function and the terminal shall print END ADDR

(b) Enter the last address to be traced and then a CR (Carriage Return) character. The terminal shall print an asterisk on the next line.

(c) Refer to Table 4-7 and enter the appropriate trace command. The terminal prints the contents of the program registers for each instruction through the selected ending address.

TABLE 4-7. Trace to Ending Address Commands

COMMAND	OPERATION
n;G	Start at the auto restart location and trace the user's program.
;G	Start at memory location n and trace the user's program.
;P	Start at present memory address and trace the user's program.

(d) Enter the characters ;T to remove the EXORciser from the trace mode. The EXORciser returns to the MAID function and the terminal prints an asterisk on the next line.

4-20 BIT PATTERN SEARCH FUNCTION

The Bit Pattern Search Function instructs the EXORciser to search a selected section of memory for a specific bit pattern. The user has the option of selecting the bits of the memory bytes to be searched by entering in hexadecimal a number to be ANDed with the byte. A high level in the number enables checking the particular bit and a low level inhibits (masks) checking of this bit. For example, the hexadecimal number FF enables the EXORciser to check the bit pattern of all eight bits in the byte while the number OF enables the EXORciser to check the bit pattern of the four least significant bits of the byte. It is assumed at the start of this function that the EXORciser is in the MAID function and the last data printed by the terminal is an asterisk. Figure 4-15 depicts a typical bit search pattern function.

```
EXBUG 1.1 MAID
*$M 23 FF
BEG ADDR 0010
END ADDR 002E 3B
*BD;W
001D BD
0033 BD
0038 BD
*
```

FIGURE 4-15. Typical Bit Pattern Search Function

4-20.1 SET SEARCH MASK

(a) Enter the character $M. The EXORciser is placed in the bit pattern search function and the terminal prints 00 on the same line.

(b) Enter in hexadecimal the bits in the byte to be searched (i.e. FF for all eight bits or OF for the four least significant bits) and then a CR (Carriage Return) character. The terminal shall print BEG ADDR nnnn on the next line.

(c) Enter in hexadecimal the beginning address of the memory section to be searched and then enter a CR character. The terminal shall print END ADDR nnnn on the next line.

4-20.2 START BIT SEARCH (n;W)

(a) Enter the bit pattern (n) to be searched for, a semicolon, and the character W. The EXORciser searches the selected portion of memory for the bit pattern and the terminal prints the address of matching bits. At the conclusion of this function, the EXORciser returns to its MAID control program and the terminal prints an asterisk.

4-21 NUMERICAL CONVERSION FUNCTION

The EXORciser has the capability of converting decimal numbers into their hexadecimal equivalent, octal numbers to their hexadecimal equivalents, and hexadecimal numbers to the decimal equivalents. It is assumed that the EXORciser is in the MAID function and the last data printed by the terminal is an asterisk. Figure 4-16 depicts typical numerical conversion functions.

```
EXBUG 1.1 MAID
*#1024=0400
*#@200=0080
*#$400=01024
```

FIGURE 4-16. Typical Numerical Conversion Functions

Decimal-to-hexadecimal conversion.

(a) Enter a pound sign (#) character and the decimal number to be converted.

(b) Enter an equal sign (=) character. The EXORciser returns to the MAID function while the terminal prints the hexadecimal equivalent and on the following line an asterisk.

Octal-to-hexadecimal conversions.

(c) Enter a pound sign (#) commercial at sign @, and the octal number to be converted.

(d) Enter an equals sign. The EXORciser returns to the MAID function while the terminal prints the hexadecimal equivalent and on the following line an asterisk.

Hexadecimal-to-decimal conversion.

(e) Enter a pound sign (#), a dollar sign ($), and the hexadecimal number to be converted.

(f) Enter an equals sign. The EXORciser returns to the MAID function while the terminal prints the decimal equivalent and on the following line an asterisk.

CHAPTER 5 THEORY OF OPERATION

5-1 INTRODUCTION

This chapter provides a block diagram description of the EXORciser (M685DT) and each of its printed circuit modules. The EXORciser as a system development tool may be configured in a variety of applications and with a variety of options. This chapter, rather than discussing each possible configuration, discusses the basic EXORciser with each otpion (except the Wirewrap Module) discussed in a supplement to this manual. The basic EXORciser comes equipped with an MPU Module (Paragraph 5-3), a Debug Module (Paragraph 5-5), the Baud Rate Module (Paragraph 5-6), and the Power Supply (Paragraph 5-7). The optional Wirewrap Module (Paragraph 5-8) is also discussed in this chapter.

The basic EXORciser and its optional modules provide the user with a system development tool for the M6800 Microcomputer Family of Parts. The user, through his configuration of the optional modules in the basic EXORciser unit, has the capability of emulating (functionally creating) a hardware prototype of his system. The user now loads his assembled program into the EXORciser and debugs his system hardware and software.

5-2 BASIC EXORciser BLOCK DIAGRAM DESCRIPTION

The basic EXORciser, as illustrated in Figure 5-1, consists of the MPU Module, the Debug Module, the Baud Rate Module, the power supply, the chassis, the EXORciser bus, and the EXbug Firmware. Each of these modules is built around the M6800 Microcomputer Family of Parts — MC6800 Microprocessing Unit (MPU), MC6820 Peripheral Interface Adapter (PIA), MCM6810 Random Access Memory (RAM), MCM6830 Read Only Memory (ROM), and MC6850 Asynchronous Communications Interface Adapter (ACIA).

The EXORciser EXbug Firmware, through its system debug and program control features, minimizes the time required to develop a user's system. The EXbug firmware provides the EXORciser with the capability to:

- Display the contents of the MPU registers at any time
- Step through the user's program one instruction at a time
- Trace through a user's program to locate problem areas
- Stop the program on a selected program step
- Abort from the user's program and return to the EXbug control program on command
- Reinitialize the EXORciser on command.

The user communicates with the EXORciser in one of two ways:

- Through a RS-232C or TTY data terminal
- Through the EXORciser front panel controls and indicators.

The data terminal permits the user to communicate directly with the EXbug Firmware. The EXORciser front panel permits the user to apply power to the EXORciser, to abort (exit) the EXORciser from a routine, and to initialize and reset the EXORciser.

The MPU Module incorporates the MC6800 Microprocessing Unit (MPU) and the system clock. This module, as illustrated, performs a dual function in that this module provides the MPU and the clock signals for both the EXORciser Debug and the user's prototype system. The MC6800 Microprocessing Unit is an 8-bit parallel processing unit capable of addressing 64k bytes of memory. In addition, the MPU addresses its input and output devices as memory. The MPU also provides the EXORciser with 72 variable length instructions and the capability of responding to real time interrupt signals.

The MPU Module controls the flow of commands, data, and addresses on the EXORciser bus. During a MPU memory read or write operation the MPU Module controls the transfer of command, status, addresses and data to the selected module by controlling the EXORciser bus.

With the exception of the EXORciser clock, the system restart, and the DBE signal (delayed); the MPU Module appears exactly like a MC6800 Microprocessing Unit with unlimited TTL drive capability.

The Debug Module provides the EXORciser with its capability to evaluate and debug the user's prototype hardware and software in an actual application. The EXbug Firmware, contained in the module's ROM memory, provides the EXORciser with its program control capabilities. The module's RAM memory is used as a scratch-pad memory for the EXbug Firmware. The EXbug Firmware enables the EXORciser to:

- Load data into the EXORciser
- Verify that the data in the EXORciser is valid
- Search a tape for a specific file
- Print the contents of the memory
- Punch (or record) the contents of the memory
- Perform the MAID (Motorola Active Interface Debug) functions.

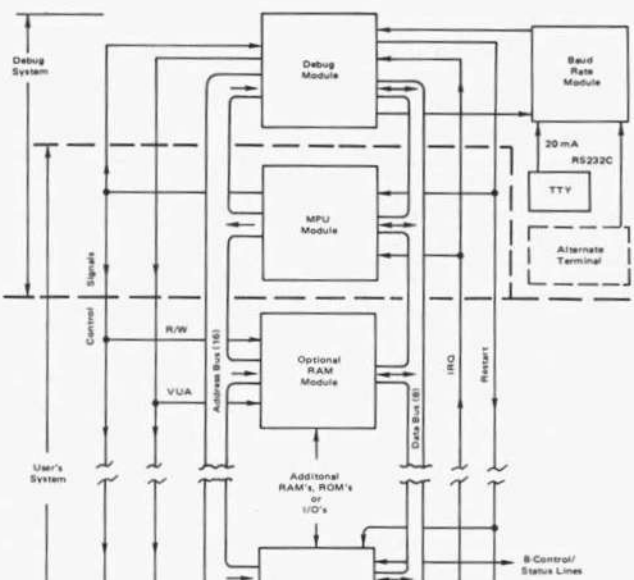


FIGURE 5-1. Basic EXORciser Block Diagram

The MAID function enables the EXORciser to:

- Examine and, if required, change the contents in a memory location
- Examine and, if required, change the contents of the MPU registers
- Calculate the offset in the relative addressing mode
- Insert, display, and remove breakpoints in the user's program
- Freerun or trace through a user's program under MAID control
- Search memory for a specific bit pattern
- Perform decimal-octal-hexadecimal conversions
- Stop the EXORciser on a selected memory address in the user's program
- Provide a oscilloscope trigger pulse at a selected memory address.

The Debug Module also incorporates the level converter circuits required to interface the EXORciser with a TTY or RS-232C data terminal.

The Baud Rate Module provides the EXORciser with eight standard baud rates — 110, 150, 300, 600, 1200, 2400, 4800, and 9600 baud. This module also interfaces the EXORciser with a TTY or RS-232C compatible data terminal.

The Power Supply provides the EXORciser with the +5 VDC, +12 VDC, and −12 VDC power sources required by the EXORciser and its modules. This power supply will support a full rack of modules.

The chassis is capable of holding 14 plug-in modules. These 14 sockets connected directly into the

EXORciser bus. The Power Supply and Baud Rate Module are not plug-in modules and are mounted directly to the EXORciser chassis.

5-3 MPU MODULE

5-3.1 GENERAL DESCRIPTION

The MPU Module serves a double function in the EXORciser. This module provides the timing and microprocessing unit for both the EXORciser and the user's prototype system. The module's timing circuit determines the EXORciser's clock frequency and provides the EXORciser with the capability to refresh dynamic memories. The module also interfaces the MC6800 Microprocessing Unit (MPU) with the EXORciser bus. The MPU performs all of the EXbug Firmware and user's program instructions. A block diagram of the MPU Module is presented in Figure 5-2 and the module's schematic diagram is illustrated in Figure 5-5.

5-3.2 BLOCK DIAGRAM DESCRIPTION

The MPU Module, as illustrated in Figure 5-2, provides the user with the option of using its internal 1 MHz crystal clock or an external clock signal. The MPU module is capable of accepting an external clock input between 100k Hz and 1 MHz. The external clock signal is applied to the EXT CLK (EXTernal CLocK) input terminals E3 and E4 of the external clock input circuit. Switch S1 selects the clock signal to be applied to the clock control circuit and to the EXORciser bus as the MEMCLK (MEMory CLocK).

The clock control circuit converts its selected input clock signal into the two non-overlapping clock signals $\phi 1$ and $\phi 2$ required by the MC6800 Microprocessing Unit and the EXORciser. This circuit also applies the $\phi 2$ clock signal to the R/W control circuit and the data bus control circuit. The clock control circuit delays the $\phi 2$ clock pulse 100 to 250 ns and applies this signal to the MC6800 Microprocessing Unit and the data bus control circuit as the DBE (Data Bus Enable) signal. This signal enables the EXORciser to transfer data on its bus during time $\phi 2$.

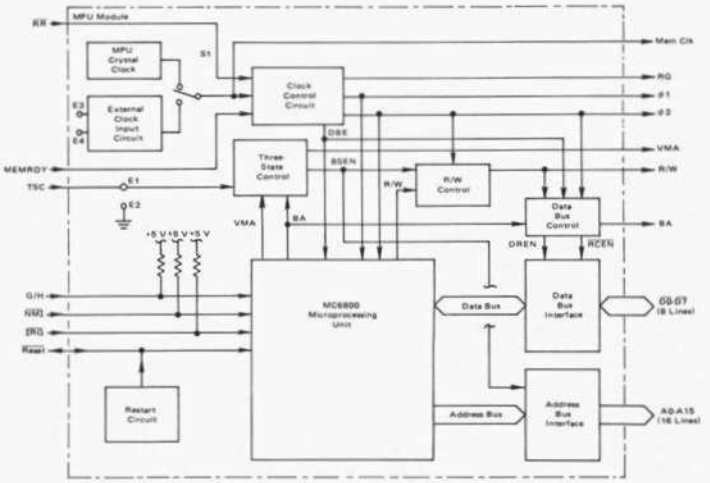


FIGURE 5-2. MPU Module Block Diagram

The clock control circuit also provides the MPU Module with a dynamic memory refresh capability. Any dynamic memory module in the EXORciser can initiate a memory refresh operation by placing a $\overline{RR}$ (Refresh Request also called $\overline{REFREQ}$) command on the EXORciser bus. Assume that a dynamic memory module has just applied a $\overline{RR}$ command to the timing control circuit. This $\overline{RR}$ command inhibits the timing control from generating its $\phi 1$ and $\phi 2$ clock signals and, at the same time, instructs the timing control circuit to generate a RG (Refresh Grant also called REF GRANT) command. The RG command instructs the requesting memory to refresh itself using the MEMCLK and remove the $\overline{RR}$ command. The memory control, on the removal of the $\overline{RR}$ command, is again enabled to generate $\phi 1$ and $\phi 2$ clock.

The MPU Module, through the MEMORY READY command, has the capability of working with slow memory modules. The MEMORY READY command on going low also inhibits the control logic from generating its $\phi 1$ and $\phi 2$ clock signals. The clock control circuit now stretches the $\phi 2$ pulse and waits for the initiating module to complete its memory operation. At the completion of this operation the initiating module returns the MEMORY CLOCK to a high level enabling the clock control circuit to again generate the $\phi 1$ and $\phi 2$ clock.

The three state control circuit, on receiving a low level TSC (Three State Control) input is enabled to process the BA (Bus Available) signal. The TSC is made a low level by either using a jumper between E1 and E2 (making it permanently low) or by receiving a low level TSC signal from the EXORciser bus (no jumper between E1 and E2). When the BA signal is present, the three state control circuit applies a BSEN (BuS ENable) signal to the R/W control circuit and to the address bus interface. This signal enables the R/W control circuit and the address bus interface to place the R/W command and the selected memory address on the EXORciser bus. A high level TSC input or the absence of the BA signal inhibits the three state control from generating a BSEN signal. The absence of the BSEN signal forces the R/W control circuit drivers and the address bus interface drivers to their off or high impedance state. That is, these drivers place a high impedance output to the EXORciser bus.

The data bus control circuit decodes its inputs and determines whether to transfer data into or from the MC6800 Microprocessing Unit. This circuit also places the BA signal on the EXORciser bus. During a MPU write operation the data bus control receives a low level R/W command and AND's this signal with the DBE ($\phi 2$ clock signal delayed 100 to 250 ns) signal and the BA signal forming a DREN (DRiver ENable) command. The DREN command instructs the data bus interface to transfer data from the MC6800 Microprocessing Unit to the EXORciser bus. During a MPU memory read operation the data bus control AND's the high level R/W command with the $\phi 2$ clock signal forming a $\overline{RCEN}$ (ReCeiver ENable) signal. The $\overline{RCEN}$ signal instructs the data bus interface to transfer data from the EXORciser bus to the MC6800 Microprocessing Unit.

The restart circuit generates a $\overline{RESET}$ signal approximately 500 ms after power is applied to the EXORciser. This module also receives a $\overline{RESET}$ signal each time the EXORciser's RESTART switch is actuated. The $\overline{RESET}$ signal instructs the MC6800 Microprocessing Unit to perform its initialization routine. This signal also resets and initializes the EXORciser.

The MC6800 Microprocessing Unit is the central processing unit for the EXORciser and the user's prototype system under development. Refer to the MC6800 Microprocessing Applications Manual for a complete explanation of the MPU's operation.

5-4 EXORciser BUS

The EXORciser bus, as illustrated in Figure 5-1, interfaces the MPU Module with the other modules being used in the EXORciser. This bus provides the EXORciser with its flexibility in being configured to meet a user's specific application.

The MPU Module controls the operation of the EXORciser bus. This module, using the bus, is capable of addressing 64k bytes of memory. The MPU Module also addresses its peripheral devices as memory. This means that the MPU Module addresses each of the EXORciser's modules as memory. During a MPU memory read or write operation the MPU Module places on the bus the address of the memory location to be exorcised. This module also transfers on the bus the commands and timing signals required to perform the assigned operation. The bus, during time $\phi 2$, provides a two-way data transfer capability between the MPU Module and the selected memory location. Table 2-6 identifies and describes each of the bus signals.

It is possible for another module to gain control of the bus. This module would place a low level G/H (Go/Halt) on the bus and monitor the BA signal. When the MPU Module completes the instruction it is performing, it generates a high level BA signal. The module requesting control of the bus now must pull the TSC line low forcing the MPU Module address bus interface drivers to their high impedance state. The requesting module has gained control of the bus until it selects to relinquish control. To relinquish control this module must remove the TSC signal and, on the next $\phi 2$ clock pulse, release the G/H line.

The user in preparing the EXORciser as a prototype of his system, sets the base memory location addresses for each of the memory and peripheral interface modules. The EXORciser bus permits the user to insert these modules in any of the EXORciser's 14 card slots and to move the module from card slot to card slot as required without changing his user's program. This bussing approach also permits the user to change the base memory address of a module without having to make any hardware design changes in his prototype system.

5-5 DEBUG MODULE

5-5.1 GENERAL DESCRIPTION

The Debug Module (refer to Figure 5-1) pro-

vides the EXORciser with the capability of evaluating and debugging a user's prototype system hardware and software. Its ROM memory contains the EXbug Firmware that provides the EXORciser with its unique system development capabilities. The module's RAM memory acts as a scratch-pad memory to the EXbug program. The EXbug Firmware enables the EXORciser to:

- Load data into the EXORciser
- Verify that the data in the EXORciser memory is valid
- Search a tape for a specific file
- Print the contents of the memory
- Punch the contents of the memory on tape (or record on a tape cassette)
- Perform the MAID (Motorola Aided Interface Debug) function.

The MAID functions enable the EXORciser to:

- Examine and, if required, change the contents in a memory location
- Examine and, if required, change the contents of the MPU registers
- Calculate the offset in the relative addressing mode
- Insert, display, and remove breakpoints in the user's program
- Freerun or trace through the user's program under MAID control
- Search the memory for a specific bit pattern
- Stop the EXORciser on a selected address in the user's program
- Provide an oscilloscope trigger pulse at a selected memory address
- Perform decimal-octal-hexadecimal conversions.

The Debug Module also incorporates the level converter circuits required to interface the EXORciser with a TTY or RS-232C data terminal.

A block diagram of the MPU is presented in Figure 5-3 and the modules schematic diagram is illustrated in Figure 5-6.

5-5.2 DEBUG MODULE BLOCK DIAGRAM DESCRIPTION

The Debug Module's block diagram description is divided into two parts. The first part (Paragraph 5-5.2.1) discusses the modules various operations and the second part (Paragraph 5-5.2.2) discusses briefly each of the circuits on the module.

5-5.2.1 Debug Module Operations

The following paragraphs discuss each of the operations performed by the Debug Module. These operations enable the EXORciser to perform its EXbug system development capabilities and are under EXbug control.

EXBUG ENABLE. The user, through the EXBUG switch on the Debug Module (refer to Figure 5-3), can select to use or to inhibit this modules EXbug Firmware and hardware capabilities. When the EXBUG switch is in its ON position, the Debug Module is enabled to perform its EXbug function and, when OFF, inhibits the module.

INITIALIZE AND RESTART OPERATION. Approximately 500 ns after power is applied to the EXORciser, the Debug Module receives a reset pulse from the MPU Module via the EXORciser bus. This RESET pulse instructs the MC6800 Microprocessing Unit on the MPU Module to perform its restart routine. The Debug Module applies this signal to the restart and master clear circuit which, in turn, generates a $\overline{MR}$ pulse. The restart and master clear circuit also generates a $\overline{MR}$ pulse each time the EXORciser RESTART switch is actuated. In this case, however, the restart and master clear circuit places a $\overline{RESET}$ pulse on the EXORciser bus to initiate the MPU Module restart routine.

The $\overline{MR}$ pulse resets and initializes the EXbug control logic, the stop-on-address decoder, the NMI-ROI counter, the NMI initiate circuit, and the restart circuit. At this time the MC6800 Microprocessor is performing its cold-start power-up routine. In the first two steps of this routine the MPU Module reads memory address $FFFE_{16}$ (for the eight MSB's) and $FFFF_{16}$ (for the eight LSB's) to get the interrupt vector address ($F559_{16}$) in the following manner. The restart circuit, on being reset, applies a high level $\overline{RAMEN}$ (RAM ENable) signal to the RAM and the PROM select circuits. This high level signal inhibits the RAM memory from being read and instructs the PROM select circuit to enable the MPU to read the PROM. The MPU Module during the reading of addresses FFFE and FFFF, reads the cold-start power-up vector address from the PROM. The data bus enable circuit, in enabling the data bus interface to transfer restart vector address from the PROM to the MPU Module, applies two DRVENB (DRiVer ENaBle) pulses to the restart circuit and the data bus interface. The restart circuit counts the number of DRVENB pulses and at the conclusion of the second pulse generates a low level $\overline{RAMEN}$ signal. This low level signal enables the MPU to use the RAM in performing the EXbug Firmware.

The advantage of actuating the RESTART switch rather than removing power from the EXORciser to gain program control is that the initialize and restart operation simulates the removal of power to the MPU and yet does not destroy the data in the static RAM memory devices used in the system.

MEMORY READ AND WRITE OPERATIONS. During a MPU memory read or write function, the Debug Module receives the 16 address bits A0 through A15, the $\phi 2$ timing signal, the VMA (Valid Memory Address) signal, and the R/W command via the EXORciser bus. During a memory write operation, this module also receives the eight data bits $\overline{D0}$ through $\overline{D7}$. The module applies the 16 address bits to the address bus

interface and the $\phi 2$ timing signal, the VMA signal, and the R/W command to the control bus interface. The address bus interface, after buffering the input address, couples the address to the various circuits used on the module. The control bus interface, after buffering its inputs, also couples its inputs and their complements to the circuits on this module.

The EXbug control logic decodes its input address bits to determine whether the MPU Module is addressing the EXbug Firmware. The EXbug Firmware uses memory addresses $F000_{16}$ through $FFFF_{16}$. If the MPU Module is not addressing the EXbug Firmware, the EXbug control logic automatically places a VUA (Valid Users Address) signal on the EXORciser bus. This VUA signal enables the other modules in the EXORciser to process the address and commands on the EXORciser bus. The EXbug control logic, on determining that the MPU Module is addressing the EXbug Firmware, reads the position of the EXBUG switch. This switch in the OFF position inhibits the EXbug control logic from generating the module's EXbug commands and instructs this circuit to generate a VUA signal for all MPU Module addresses. In the ON position, however, the EXBUG switch enables the EXbug control logic to generate the module's EXbug commands and inhibits this circuit from generating a VUA signal during the execution of the EXbug Firmware. The using of the VUA signal rather than the VMA signal to enable the optional EXORciser Modules permits the user to operate his EXORciser with partially decoded systems (not using all 16 address lines) and not reading two addresses at the same time.

The EXbug control logic, on determining that the MPU Module is addressing the EXbug program, further decodes the address bits to determine whether the MPU Module is addressing the Debug Module's RAM, ROM, PROM, stop-on-address circuits PIA, or ACIA circuits. Table 5-1 depicts the addresses, selected circuits, and EXbug enabling commands.

TABLE 5-1. EXbug Firmware Memory Addresses

Memory Addresses To	From	Selected Circuit	Enabling Command
F000	F3FF	ROM1	FVMA
F400	F7FF	ROM2	FVMA
F800	FBFF	ROM3	FVMA
FCF4	FCF7	ACIA	IOEN
FCF8	FCFB	PIA	IOEN
FCFC	FCFF	PROM	IOEN
FF00	FF7F	RAM6	RAMS
FF80	FFFF	RAM7	RAMS

During an EXbug memory read operation, the control bus interface receives a high level R/W command and applies this command with its complement to the circuits as illustrated in Figure 5-3. The EXbug control logic AND's the R/W command with the $\phi 2$ timing signal and applies the resulting $\phi 2 \cdot$ R/W signal to the ROM and the data bus enable circuit. Now, while the MPU is reading the selected memory location, the $\phi 2 \cdot$ R/W pulse instructs the data bus enable circuit to generate a DRVENB signal. This signal instructs the data bus interface to transfer data from the selected memory location on the Debug Module to the MPU Module via the EXORciser bus.

During a memory write operation of EXbug the control bus interface receives a low level R/W command and applies this command with its complement to the circuits illustrated in Figure 5-3. The EXbug control logic is inhibited by the R/W pulse from generating a $\phi 2 \cdot$ R/W pulse. Now, as the MPU Module writes into the selected memory location in the RAM, ACIA, or PIA in the stop-on-address circuit, the data bus enable circuit is enabled and applies a $\overline{\text{RECENB}}$ signal to the data bus interface circuit. This signal enables the data bus interface to transfer data from the EXORciser bus to the memory location being addressed.

ABORT FUNCTION. The user, on pressing the EXORciser ABORT switch, instructs the EXORciser to abort or exit from the program it is running and return to the EXbug control program. Each time the ABORT switch is actuated, the abort flip-flop applies an $\overline{\text{ABT}}$ (ABorT) signal to the NMI initiate circuit and to the MC6820 Peripheral Interface Adapter (PIA) in the stop-on-address decoder. The NMI initiate circuit, through its NMI (Non-Maskable Interrupt) command, initiates a MC6800 MPU non-maskable interrupt routine. The $\overline{\text{NMI}}$ command is present as long as the user holds the ABORT switch in. The MC6800 MPU, on receiving a NMI command, completes the instruction it is performing and then jumps to its non-maskable interrupt routine. In this interrupt routine the MPU stores its register contents on the stack and then polls the MC6820 PIA in the stop-on-address decoder to determine the operation initiating the interrupt. In this case, the $\overline{\text{ABT}}$ input to the MC6820 PIA instructs the MPU Module to print the statement "ABORTED AT" and the MPU register data stored on the stack. On completing the printing operation, the MPU Module goes to the EXbug control program.

STOP-ON-ADDRESS/SCOPE TRIGGER FUNCTION. The user in selecting the stop-on-address or scope trigger function loads an address into the EXORciser. The MPU Module, using two memory write operations, loads this address into the MC6820 PIA in the stop-on-address decoder. The eight least significant bits of the address are loaded into side A and the eight most significant bits are loaded into side B of the MC6820 PIA.

Now, each time the EXORciser performs an operation, the stop-on-address decoder compares the address on the bus with the address stored in its MC6820 PIA. When these two addresses compare, the stop-on-address decoder applies a $\overline{\text{SAD}}$ (Stop on ADdress) pulse to the SCOPE/STOP-ON-ADDRESS switch, S2. This switch in the SCOPE position applies the $\overline{\text{SAD}}$ pulse to the SCOPE TRIGGER jack. In the STOP-ON-ADDRESS position the switch applies the $\overline{\text{SAD}}$ pulse to the NMI initiate circuit and to the MC6820 PIA in the stop-on-

address decoder. The NMI initiate circuit through its $\overline{\text{NMI}}$ command, initiates a MC6800 MPU non-maskable interrupt routine and enables the NMI-ROI counter to count. The NMI-ROI counter applies two pulses to the NMI initiate circuit — one at the sixteenth $\phi 2$ clock pulse after the $\overline{\text{NMI}}$ was initiated and the second at the thirty-second $\phi 2$ clock pulse after the $\overline{\text{NMI}}$ was initiated. At the conclusion of the thirty-second $\phi 2$ clock pulse, the NMI initiate circuit removes the $\overline{\text{NMI}}$ command and disables the NMI-ROI counter. The MC6800 MPU, on receiving a $\overline{\text{NMI}}$ command, completes the instruction it is performing and jumps to its non-maskable interrupt routine. The MPU in its NMI routine stores the MPU register contents on the stack and then polls the MC6820 PIA in the stop-on-address decoder. In this case, the SAD input to the MC6820 PIA instructs the MPU Module to print the statement "STOP-ON-ADDRESS" and the MPU register contents on the stack. On completing the printing operation, the MPU Module goes to the EXbug control program. It should be noted that the MPU completes the instruction it is processing before it processes the $\overline{\text{NMI}}$ command; therefore, the MPU may not stop on the select address but on the address of the next instruction.

RUN-ONE-INSTRUCTION FUNCTION. The user in the MAID function has the capability of instructing the EXORciser to run a predetermined number of instructions in either tracing through a user's program or free running through the user's program. The MPU Module on receiving the run-one-instruction input from the user, loads the run-one-instruction command into the stop-on-address MC6820 PIA. The MC6820 PIA on receiving this instruction, applies a $\overline{\text{RINST}}$ (Run-one-INSTruction) signal to the ROI (Run One Instruction) pulse network. This network converts the $\overline{\text{RINST}}$ signal to a 3.5ns pulse. This pulse loads the binary number 1011 (decimal 11) into the NMI-ROI counter. At the conclusion of the eleventh $\phi 2$ clock pulse, the counter instructs the NMI circuit to initiate a non-maskable interrupt routine. During the 11 $\phi 2$ clock pulse period, the MPU Module transfers the contents of the user register stored on the stack back into the MC6800 MPU registers and performs the next instruction in the user's program. The NMI initiate circuit and NMI-ROI counter now generate a 32 $\phi 2$ long $\overline{\text{NMI}}$ pulse. In the non-maskable interrupt routine the MPU Module stores the contents in the MPU register on the stack and then polls the MC6820 PIA.

If the EXORciser was instructed to run this instruction in the freerunning mode, the MPU Module goes directly to the MAID control program. If, on the other hand, the EXORciser was instructed to trace through this program, the MPU Module prints the register contents on the stack and then goes to the MAID control program.

40.*DATA TERMINAL INTERFACE*. The data terminal interface circuit, working in conjunction with the Baud Rate Module provides a two-way transfer of data between EXORciser and the selected data terminal. The user in preparing the EXORciser for operation, selects the data transfer baud rate between the EXORciser and the data terminal to be used. The Baud Rate Module applies the BIT RATE clock and the 134.49 Hz signal to the Debug Module. The BIT RATE clock is 16 times the data transfer rate. The phase detector compares the 134.49 Hz input with the BIT RATE clock and determines whether the data being transferred requires one or two stop bits. This circuit in comparing the 134.49 signal with a 110 baud rate input, forms a high level SBS (Stop Bit Select) signal. This signal is used as address bit A2 on the PROM. In comparing the seven other baud rates (all above 135 Hz) used in the EXORciser with the 134.49 Hz signal, the phase detector generates a low level SBS signal. The EXORciser in preparing to transfer data, reads the PROM to determine whether to insert one or two stop bits on the data being transferred. This determination is made by the SBS signal addressing the PROM. The EXORciser now loads a control word into the ACIA instructing it to insert the appropriate number of stop bits.

When a TTY data terminal is connected to the EXORciser, the Debug Module receives a low level TTY signal. This signal enables the receive data select circuit to process the SERIAL DATA input from a TTY data terminal via the Baud Rate Module.

When the EXORciser is interfacing with a RS-232C data terminal, the RS-232C control interface circuit receives a high level DATA TERM READY signal. After double inverting this signal, the RS-232C control interface circuit transfers this signal to the Baud Rate Module as the DATA SET READY signal. The Baud Rate Module transfers this signal to the RS-232C data terminal as the DATA SET READY, CARRIER DETECT, and CLEAR TO SEND signals.

The EXORciser in transferring a character to the data terminal, loads in parallel the character into the MC6850 Asynchronous Communications Interface Adapter (ACIA). The MC6850 ACIA reformats this data; adds the start bit and the appropriate number of stop bits; and serial transfers the character to the RS-232C level converter at the BIT RATE clock frequency. The RS-232C level converter converts its TTL compatible input to the RS-232C compatible signal level. The TRANSMIT DATA output of this circuit is applied to the TTY isolator and level converter circuit and through the Baud Rate Module to a RS-232C data terminal (if used). The TTY converter at this time is receiving the READER CONTROL signal from the ACIA and the TRANSMIT DATA input. These two signals are optically isolated and then converted to be TTY compatible. The TTY level converter transfers the READER CONTROL and the SERIAL DATA OUT signals through the Baud Rate Module to the TTY data terminal. The READER CONTROL signal is used to control the operation of a modified TTY data terminal.

When the user enters a character into the TTY data terminal, this character is coupled through the Baud Rate Module to the TTY isolator and level converter on the SERIAL DATA INPUT line. The isolator and level

converter circuit optically isolates this input, converts it to be TTL compatible, and applies this signal to the receive data select circuit. This circuit also is receiving a low level TTY input and transfers the serial input data into the MC6850 ACIA. The MC6850 ACIA serially accepts the character, examines the bit for parity, overflow, and format errors; strips off the stop and start bits; reformates the data into a parallel format; and sets the interrupt flag. The EXORciser in its EXbug program, polls the ACIA to see if it has accepted a character. On determining that the MC6850 has accepted a character, the MPU Module reads the MC6850 ACIA and transfers the character to the MC6800 MPU.

When the user enters a character into the RS-232C data terminal, this character is coupled through the Baud Rate Module to the RS-232C level converter. This circuit converts this input to be TTL compatible and applies this signal to the receive data select circuit. The receive data select circuit at this time is also receiving a high level TTY input and transfers the serial input data to the MC6850 ACIA. The MC6850 processes this input in the same manner as the TTY data terminal input.

TRACE DURING IRQ OPERATION. The EXORciser has the capability in the trace mode of tracing through a maskable interrupt routine. Assume that the user is tracing through his program and a maskable interrupt routine (IRQ) is initiated. The control bus receives a IRQ (Interrupt ReQuest) command from the EXORciser bus and applies this command to the control bus interface. After buffering, the control bus interface applies the IRQ command to the PROM as address input A3. The EXORciser now uses the commands in the PROM to trace through the IRQ routine. At the conclusion of the IRQ routine, the IRQ command is removed and the EXORciser continues its tracing through the user's program.

5-5.2.2 Debug Module Circuits

The following paragraphs discuss the function of each of the circuits on the Debug Module.

ADDRESS BUS INTERFACE. The address bus interface circuit buffers the 16 address inputs A0 through A15 and applies these signals with their complements to the Debug Module address bus.

ABORT FLIP-FLOP. The abort flip-flop circuit receives a dual input from the ABORT switch on the EXORciser front panel. The prime function of this circuit is to remove the switch bounce as it initiates a non-maskable interrupt routine.

CONTROL BUS INTERFACE. The control bus interface circuit buffers its four input commands and applies these inputs to the appropriate circuits on the Debug Module as illustrated in Figure 5-3. This circuit also applies the complement of its VUA, R/W, and $\phi 2$ inputs throughout the module.

RESET AND MASTER CLEAR CIRCUIT. The reset and master clear circuit is used to reset the Debug Module and, when the EXORciser's RESTART switch is actuated, resets and initializes the entire EXORciser. This circuit accepts the RESTART input, removes the switch bounce from the input and then dot AND's this input onto the EXORciser RESET line. Now, each time the RESET line is actuated — either by the RESTART switch or the MPU Module restart circuit — the reset and master clear circuit through its $\overline{MR}$ (Master Clear) signal, resets the Debug Module circuitry.

EXbug CONTROL LOGIC. The EXbug control logic, through its decoding of the address lines, EXBUG switch input, and command inputs, controls the operation of the Debug Module. Paragraph 5-5.2.1 discusses each of these operations.

STOP-ON-ADDRESS DECODER. The stop-on-address decoder consists of a MC6820 Peripheral Interface Adapter (PIA) and a 16-bit comparator circuit. The operation of the PIA is discussed in the M6800 Microprocessor Applications Manual. The control registers in the PIA are used in the stop-on-address, run-one-instruction, and abort operations. The 16-bit comparator circuit compares the 16 bits on the address bus with the 16 PIA data lines and, when they compare, generates a $\overline{SAD}$ (Stop-on-ADdress) signal. The $\overline{SAD}$ signal is used as a scope trigger pulse or to initiate a non-maskable interrupt routine.

ROI PULSE NETWORK. The ROI (Run-One-Instruction) pulse network circuit converts its $\overline{RINST}$ (Run-one-INSTruction) input to a 35ns pulse. This pulse is applied to the NMI-ROI counter.

NMI-ROI COUNTER. The NMI-ROI counter is a 4-stage, 16-bit counter used to control the timing of a run-one-instruction operation and to determine the time the $\overline{NMI}$ command from the NMI initiate circuit is present — 32 $\phi 2$ clock pulses. Each time the reset and master clear circuit receives a RESET signal or an input from the RESTART switch, this circuit, through its $\overline{MR}$ signal, resets the NMI-ROI counter to an all zero count.

Each time the NMI initiate circuit receives an instruction to initiate a non-maskable interrupt routine, this circuit enables the NMI-ROI counter to count. The counter records 16 clock pulses and then clocks the NMI initiate circuit. The counter records 16 addition clock pulses and clocks the NMI initiate circuit a second time. At the conclusion of this second clock pulse, the $\overline{NMI}$ initiate circuit disables the counter and removes the $\overline{NMI}$ instruction. The counter now remains at its zero count until it gets its next input command.

In a run-one-instruction operation, the NMI-ROI counter receives a RINST pulse from the ROI pulse network. This pulse loads the binary number 1011 into the NMI-ROI counter. This counter, at the conclusion of the eleventh $\phi 2$ timing pulse, is at count 0000 and instructs the NMI initiate circuit through its clock pulse to initiate a non-maskable interrupt routine. The counter now controls the NMI initiate circuit operation as described in the previous paragraph.

NMI INITIATE CIRCUIT. The NMI initiate circuit consists primarily of a two-stage, two-bit counter and a bus driver circuit. This circuit initiates a non-maskable interrupt routine each time it receives an $\overline{ABORT}$, $\overline{SAD}$, or $\overline{INT}$ command. On receiving one of

these inputs, the circuit generates an NMI command and transfers an enabling signal to the NMI-ROI counter. The NMI-ROI counter clocks the NMI initiate circuit twice. At the conclusion of the second clock pulse from the counter, the NMI initiate circuit removes the $\overline{\text{NMI}}$ command and the NMI-ROI counter enabling signal.

ROM MEMORY. The ROM memory, consisting of three preprogrammed MCM6830 1024 × 8 bit ROM devices, stores the EXbug Firmware. The EXbug control logic, through its decoding of the address bits and the EXBUG switch, determines when the MPU is addressing the ROM memory. This memory uses addresses F000 through FBFF.

RAM MEMORY. The RAM memory, consisting of two MCM6810 128 × 8 bit RAM devices, is a scratch-pad memory for the EXbug program. The EXbug control circuit determines when the MPU is addressing this memory. During a restart operation, the restart circuit, through its $\overline{\text{RAMEN}}$ signal, inhibits the RAM memory from being read.

PROM SELECT. The PROM select circuit determines when the PROM memory is to be read. The restart circuit, during a restart operation, instructs the PROM select circuit to enable the PROM memory and read the two byte cold-start interrupt vector address located at memory locations FFFE and FFFF. The PROM select circuit also enables the PROM memory when the MPU Module is reading addresses FCFC through FCFF.

PROM MEMORY. The PROM memory contains the cold-start power-up vector address, the number of stop bits to be used in the data transfer with the selected data terminal, and the instructions to be used in tracing through a user's program interrupt routine.

ACIA. The MC6850 Asynchronouc Communication Interface Adapter (ACIA) provides the parallel-to-serial and serial-to-parallel data conversions required to transfer data to and from the selected data terminal. The circuit also, in transferring data to the selected data terminal, inserts the start bit and the required stop bits on each character. This circuit tests each character it receives from the selected data terminal for formatting and parity errors. For further information about the operation of the MC6850 ACIA, refer to the M6800 Microprocessor Applications Manual.

STOP-BIT PHASE DETECTOR. The stop bit phase detector compares the BIT RATE clock signal with the 134.49 Hz signal to determine whether the data being transferred is using one or two stop bits. The stop-bit phase detector generates a high level SBS signal when the BIT RATE clock signal is at a frequency lower than the 134.49 Hz signal and a low level when it is at a higher frequency.

TTY ISOLATOR AND LEVEL CONVERTER. The TTY isolator and level converter optically isolates the TTY data terminal from the EXORciser and performs the voltage level conversions required to interface the EXORciser with the TTY data terminal.

RS-232C LEVEL CONVERTER. The RS-232C level converter performs the voltage level conversions required to interface the EXORciser with a RS-232C data terminal.

RECEIVE DATA SELECT CIRCUIT. The receive data select circuit, on receiving a low level TTY signal is enabled to transfer the SERIAL DATA IN input from the TTY isolator and level converter to the ACIA. The TTY signal is a low level when the EXORciser is interfacing with a TTY data terminal. The TTY input goes high when the EXORciser is interfacing with a RS-232C data terminal. The receive data select circuit now transfers the input data from the RS-232C level converter to the ACIA.

RS-232C CONTROL INTERFACE. The RS-232C control interface enables the EXORciser to interface with RS-232C data terminal by generating the required data transfer commands. This circuit double inverts the DATA TERM READY signal and with the Baud Rate Module, transfers to the data terminal its input signal as the CARRIER DETECT, CLEAR TO SEND, and DATA SET READY commands.

RESTART CIRCUIT. The restart circuit is used during the MPU's initialization and reset operation to load the vector address of the cold-start, power-up routine into the MPU. This circuit, on receiving an $\overline{\text{MR}}$ signal is reset and applies a high level $\overline{\text{RAMEN}}$ signal to the RAM memory and the PROM select. This signal while high, inhibits the MPU from reading the RAM memory and instructs the PROM select to enable the PROM memory to be read. The restart circuit is clocked by the data bus enable circuit as the MPU reads addresses FFFE and FFFF of the PROM. At the conclusion of the second read operation, the restart circuit removes the high level $\overline{\text{RAMEN}}$ signal. The low level $\overline{\text{RAMEN}}$ signal enables the MPU to address the RAM and places the PROM select circuit under the control of the EXbug control circuit.

DATA BUS ENABLE. The data bus enable circuit through its decoding of the R/W command, determines the direction of data flow on the Debug Module's data bus. During a MPU memory read operation this circuit applies a DRVENB signal to the data bus interface circuit. The DRVENB signal instructs the data bus to transfer data from the selected memory location on the Debug Module to the MPU. During a memory write operation, the data bus enable circuit, through its $\overline{\text{RECENB}}$ signal, enables the data bus interface to transfer data from the MPU to the selected memory address on the Debug Module.

DATA BUS INTERFACE. The data bus interface provides a two-way transfer of data between the EXORciser's bus and the circuits on the Debug Module. When the driver and receivers are not enabled to transfer data, they provide a high impedance output to their respective busses.

5-6 BAUD RATE MODULE

5-6.1 GENERAL DESCRIPTION

The Baud Rate Module provides the EXOR-

ciser with eight standard baud rates between 110 and 9600 baud. The baud rates are 110, 150, 300, 600, 1200, 2400, 4800, and 9600. This module also interfaces the Debug Module with the EXORciser front panel and with a TTY or RS-232C data terminal. A block diagram of the Baud Rate Module is presented in Figure 5-4 and the schematic diagram in Figure 5-7.

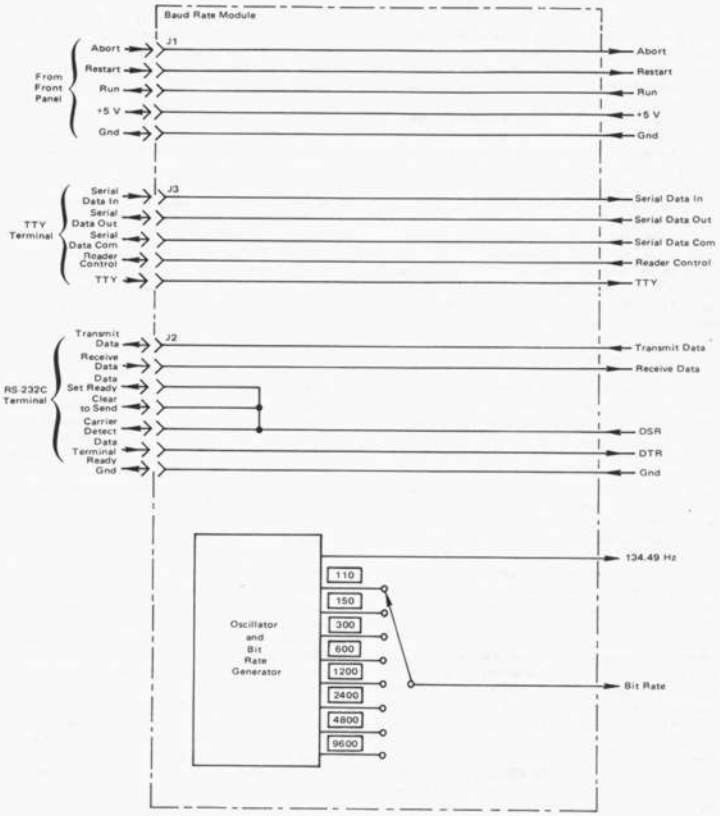


FIGURE 5-4. Baud Rate Module Block Diagram

5-6.2 BLOCK DIAGRAM DESCRIPTION

The Baud Rate module consists of the oscillator and bit rate generator, the BAUD RATE switch, and interconnection leads between the EXORciser front panel, a TTY data terminal, and a RS-232C data terminal.

The oscillator and bit rate generator circuit generates the eight baud rate frequencies used in the EXORciser. These eight frequencies are applied to the BAUD RATE switch. The user through the BAUD RATE switch selects the frequency to be transferred to the Debug Module. It should be noted that the bit rate clock signal is sixteen times the baud rate frequency data bit transfer rate. The oscillator and bit rate generator also applies the 134.49 Hz to the Debug Module. This Debug Module uses this signal to determine the number of stop bits to be used in the data transfer operation.

5-7 POWER SUPPLY

The power supply provides the EXORciser with +5 VDC, +12 VDC, and −12 VDC. The +5 VDC is used by the EXORcisers logic circuits and the +12 VDC and −12 VDC are used to interface the EXORciser with the selected data terminal. These three voltages also may be used in the construction of the user's customized circuitry.

The power supply provides both overload and short protection without damage to the power supply. In addition, the power supply provides overvoltage protection. In the case of power failure, the power supply provides all dc voltages for a minimum of 16 ms. This time interval provides the MC6800 MPU sufficient time to store its register contents before power is removed. Now, if the EXORciser is using a battery backup, the data in memory can be preserved.

5-8 WIREWRAP MODULE

The optional Wirewrap Module allows the user to construct any customized circuitry required in his prototype system. This module incorporates the power bus and ground bus printed wiring, and has standard pin spacing for 75, 14 pin wirewrap sockets and provisions for two 50-pin flatribbon connectors.

APPENDIX A
HEXADECIMAL NUMBERING SYSTEM

The M6800 Microcomputer Family of Parts uses the hexadecimal numbering system to express its address, instruction set, and register contents. The hexadecimal numbering system has a base (radix) of 16 and uses 16 distinct symbols. Table A-1 lists these symbols, their binary equivalent, and their decimal equivalent.

The following paragraphs discuss binary-to-hexadecimal conversion, hexadecimal-to-binary conversion, decimal-to-hexadecimal conversion, and hexadecimal-to-decimal conversion.

A-1 BINARY-TO-HEXADECIMAL CONVERSION

Use the following procedures to convert a binary number to its hexadecimal equivalent. In this procedure we will convert the address 0001001011011001 to its hexadecimal equivalent.

(a) Starting at the hexadecimal point and working to the left, divide the binary digits into groups of four. In breaking down the number 0001001011011001, we have

0001, 0010, 1101, 1001

(b) Referring to Table A-1, convert each group of four bits to their hexadecimal equivalent. In converting 0001001011011001 we get

0001 = 1, 0010 = 2, 1101 = D, and 1001 = 9

(c) Combine the hexadecimal digits from LSB to MSB. In combining the digits we get

$0001001011011001_2 = 12D9_{16}$

A-2 HEXADECIMAL-TO-BINARY CONVERSION

Use the following procedures to convert a hexadecimal number to its binary equivalent. In this procedure we will convert the hexadecimal number $E38F_{16}$ to its binary equivalent.

(a) Referring to Table A-1, convert each hexadecimal digit to its four binary equivalent. In converting $E38F_{16}$ we get

E = 1110, 3 = 0011, 8 = 1000, and F = 1111

(b) Combine the binary equivalent of the hexadecimal digits to form the binary number. In combining the numbers we get

$E38F_{16} = 1110001110001111$

TABLE A-1. Binary, Decimal, and Hexadecimal Equivalents

Binary Number	Decimal Number	Hexadecimal Number
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	10	A
1011	11	B
1100	12	C
1101	13	D
1110	14	E
1111	15	F

A-3 DECIMAL-TO-HEXADECIMAL CONVERSION

Use the following procedures to convert a decimal number to its hexadecimal equivalent. In this procedure we will convert the decimal number 235 to its hexadecimal equivalent.

(a) Divide the decimal number by 16 and record the remainder. In dividing 235 by 16 we get

16 235
14 with a remainder of 11.

(b) Repeat step a until 16 can no longer be divided into the decimal number. In dividing 14 by 16 we get

16 14
0 with a remainder of 14.

(c) Referring to Table A-1, convert the remainders to their hexadecimal equivalents. Converting 11 and 14 we get

11 = B, and 14 = E.

(d) Arrange the remainders in the order indicated below. The remainders form the hexadecimal equivalent of the binary number.

16 235
16 14 with remainder of 11 = B LSB
0 with remainder of 14 = E MSB

$235_{10} = EB_{16}$

A-4 HEXADECIMAL-TO-DECIMAL CONVERSION

Use the following procedures to convert a hexadecimal number ot its binary equivalent. In this procedure we will convert $80FB_{16}$ to its decimal equivalent.

(a) Multiply each hexadecimal number by its position value. In multiplying the numbers in $80FB_{16}$ by their position value we get

$8 \times 16^3 = 8 \times 16^3 = 8 \times 4096 = 32768$
$0 \times 16^2 = 0 \times 16^2 = 0 \times 256 = 0$
$F \times 16^1 = 15 \times 16^1 = 15 \times 16 = 240$
$B \times 16^0 = 11 \times 16^0 = 11 \times 1 = 11$

(b) Add the values found in step a. The sum is the decimal equivalent of the hexadecimal number. In adding the power values times $80FB_{16}$ we get

$32768 + 0 + 240 + 11 = 33019$
or $80FB_{16} = 33019_{10}$

APPENDIX B
TTY MODIFICATIONS

B-1 INTRODUCTION

This appendix discusses configuring a TTY terminal for full-duplex 20 mA neutral loop current operation and for modifying a TTY terminal for automatic reader/punch control. This appendix is intended for general information and it is recommended that the user refer the terminal manufacturer's documentation to make these changes.

BC-2 FULL-DUPLEX 20 mA LOOP CURRENT CONFIGURATION

Figure B-1 depicts the connections to be used for a full-duplex 20 mA neutral loop current TTY terminal. Refer to this figure and make the necessary following connections on your terminal.

(a) Connect the white/blue wire to terminal 5.

(b) Connect the brown/yellow wire to terminal 5.

(c) Connect the violet wire to terminal 9.

(d) Move the blue from the 750 ohm tap to the 1450 ohm tap on terminal power resistor. This resistor is located behind the terminal's power supply.

B-3 AUTOMATIC READER/PUNCH CONTROL MODIFICATION

The EXORciser provides a command signal to be used for automatic reader/punch of a modified TTY terminal (refer to Figure B-2). Modify your terminal as follows:

(a) Install a 1200 ohm reed relay into the teletypewriter.

(b) Connect the relay contacts such that when the LOCAL-OFF-LINE switch is in LINE and the relay energizes, the TTY will operate.

(c) Connect the relay coil such that one side is connected through the cable to J3-2 and the other side of the coil to J3-4.

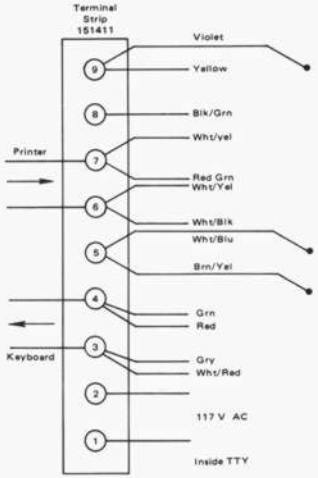


FIGURE B-1. TTY Full-Duplex 20 mA Neutral Loop Terminal Connections

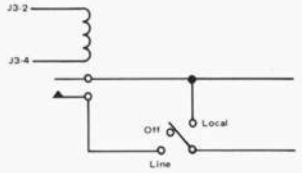


FIGURE B-2. TTY Automatic Reader/Punch Control Modification