SHOOTER

EPROM Programmer

User's Manual

LOGICAL

LOGICAL DEVICES, INC.

1321 E Northwest 65th Place,
Fort Lauderdale, FL 33309
305-974-0967

Return Programmer → *Modem ^E

Return *Modem → CP/M CPM <RET>

Zen Link

Config.

COM Port	2 - COM2
BAUD	1200
WORD	8 - bits
Parity	None
Stop Bit	2 - Bits
Terminal	Glass - TTY
XON/XOFF	Disabled
Local CD	Disabled
Host	Disabled
Echo	Enabled

Release Information

This manual describes the operation of Logical Devices' SHOOTER (Prompro-4) EPROM Programmer. Logical Devices, Inc. reserves the right to modify, ammend, or in any other way change the contents and/or product described herein, at any time, without notification.

The information contained in this manual has been reviewed for accuracy, clarity, and completeness. Logical Devices, Inc. will not be responsible for errors, misinterpretation of contents, or use inconsistant with the information contained in this manual.

Please report, in writing, any errors to Logical Devices, Inc., 1321 NW 65th Place, Fort Lauderdale, FL 33309. Logical Devices, Inc. reserves the right to use and distribute any information supplied, without obligation.

TABLE OF CONTENTS

APPENDICIES

1.0 UNPACKING

1. Carefully unpack unit from the shipping container and inspect for possible shipping damage.

2. Check for the following items:

   a) SHOOTER EPROM programmer
   b) Set of 10 configurators (for 2716, 2732, 2732A, 2532, 2764, 2764A, 27128, 27128A, 27256, 27256 [21 volt])
   c) Warranty registration
   d) Serial communication cable (without RS-232 connector)

2.Ø OPERATION

The SHOOTER programmer is a complete stand-alone unit with an RS232 interface capability. All operating controls and indicators are contained on the front panel and at the back of the unit as shown in Figure 2-1. There are three basic modes of operation: stand-alone, terminal controlled, and computer controlled.

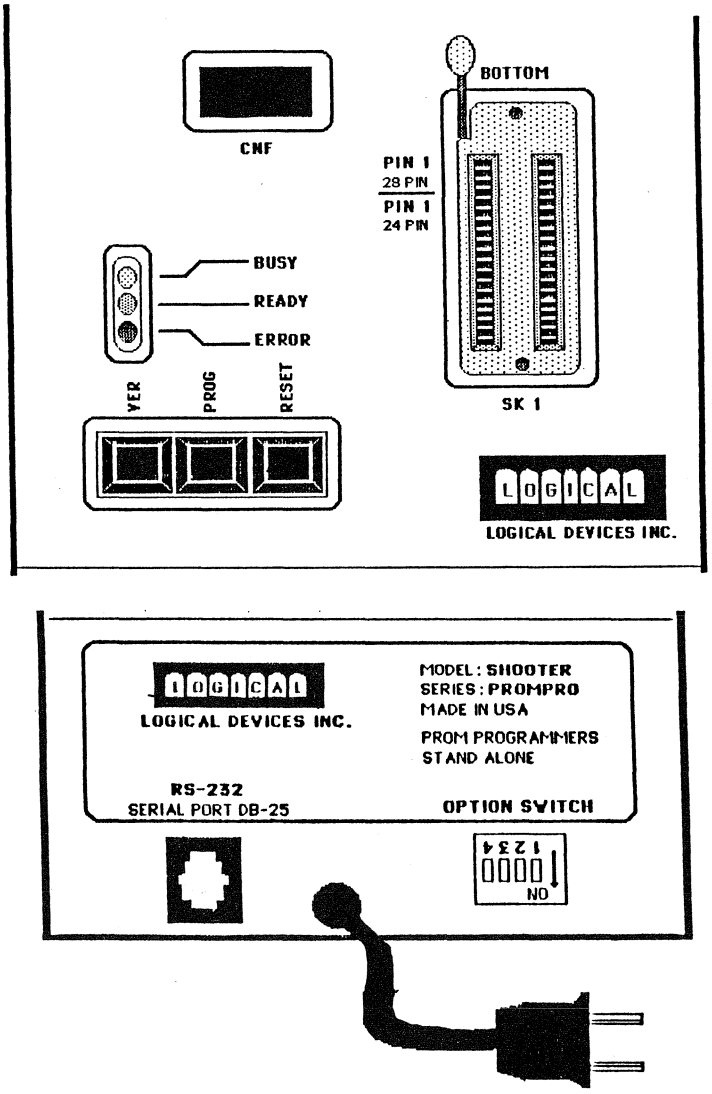


Figure 2-1. SHOOTER Front Panel and Rear Panel Controls and Indicators.

2.1 STAND ALONE MODE

In the stand-alone mode of operation, you copy and verify EPROMs using the front panel function switches. There is no need for a computer or terminal interface in this mode of operation.

NOTE

220 volt operation requires a 220 volt modification option.

1. Plug the unit into the appropriate AC outlet. The front panel status LEDs will light in sequence (red - ERROR, yellow - BUSY, green - READY). The red will flash briefly, the yellow will flash for a short while, and then the green LED will stay on to indicate that the unit is ready to accept a command.

2. Refer to Section 5.0 to choose the proper configurator plug. With the white dot to the left, insert the proper Configurator plug in the CNF socket.

3. Insert the master EPROM (the one you wish to copy) into the ZIF socket. Align the EPROM so that pin 1 is in the upper left corner of the ZIF socket (the same side as the ZIF handle) and the ground pin (pin 12 of 24-pin EPROMS, pin 14 of 28-pin EPROMS) is in pin 14 of the ZIF socket. Refer to Appendix E for additional EPROM insertion instructions.

4. Press the front panel RESET switch to reset the system and automatically read the entire contents of the master EPROM into the programmer RAM buffer.

NOTE

The front panel LED status indicators will light in sequence as in the power-up procedure.

5. Remove the master EPROM and replace it with an erased EPROM of the same type.

NOTE

To verify that an EPROM is properly erased in the stand-alone mode:

a. Insert a known blank EPROM in the unit and press the RESET switch.
b. Insert the EPROM in question into the unit.
c. Press the VERIFY switch. If the red LED status indicator lights, the EPROM is not completely erased.

6. Be sure both the EPROM and the configurator plug are inserted correctly in their respective sockets. This is a very important step to prevent damage to the device.

----- CAUTION -----

Before continuing, be sure all previous steps have been performed correctly or you may damage the device.

7. Depress the PROG switch on the front panel (the yellow BUSY indicator will remain on during programming). When the program cycle is complete, the green READY indicator will light. If the EPROM does not program properly, the red ERROR indicator will light.

8. To insure that every location has been programmed properly, initiate the Verify feature by pressing the VERIFY switch on the front panel. This will initiate a comparison of every data byte in the EPROM against the programmer RAM. If there is any discrepancy, the red ERROR indicator will light.

2.2 TERMINAL MODE/COMPUTER MODE

NOTE

Generally, the serial cabling to a computer is different from the serial cabling to a CRT terminal. Refer to Section 6.0 for wiring details.

2.2.1 Terminal

SHOOTER has the ability to operate directly with a terminal. An internal communications program allows you to use the terminal to control SHOOTER operation. Refer to Section 6.0 for interface connection and baud rate setting information.

1. Connect the unit to a serial terminal or a computer through the serial communication connector located on the back of the unit. Set the dip switches on the rear panel for the the desired baud rate.

2. Upon power up, SHOOTER sends a command menu to the terminal followed by a * prompt. A command may be entered after the prompt appears. Refer to Section 3.Ø for detailed descriptions of the following commands:

C = CHECKSUM	H = HEX LOAD
F = FILL RAM	G = HEX DUMP
N = BLANK CHECK	L = BINARY LOAD
P = PROGRAM	K = BINARY DUMP
R = READ EPROM	T = DOWNLOAD
V = VERIFY	U = UPLOAD
X = EXAMINE	W = UPLOAD WITH WAIT
? = MENU	O = OFFSET xx
I = INTEL	
M = MOTOROLA	

NOTE

An * appears beside either Intel Hex or Motorola Hex to indicate in which mode SHOOTER is currently.

NOTE

You may display the menu at any time by typing a question mark.

4. After completing any command, SHOOTER transmits either the character sequence ^F*EOJ* or ^U*EOJ*. The character ^ preceding any character indicates the control code for that character. A ^F (HEX Ø6) indicates an ACK (Acknowledge) meaning that all went well in the previous instruction. A ^U (HEX 15) transmitted indicates a NAK (Negative Acknowledge) meaning that an error was encountered on the previous instruction. The codes $Ø6 or $15 are typically non-displaying characters that would only be used when connected to a host computer.

NOTE

^U*EOJ* transmission sequence means a control U, followed by ASCII characters *EOJ* is sent.

5. Files may be uploaded to the terminal screen from the SHOOTER for viewing. Files may also be sent directly to the SHOOTER RAM by manually keying in the data from the terminal keyboard.

2.2.2 Computer

The SHOOTER also has the ability to communicate with a computer, in one of two ways, if the computer is executing the appropriate software.

1. COMPUTER EMULATING A TERMINAL (Host terminal) - To make your computer function as a terminal, you must instruct your computer via a software program. This type of software program is generally referred to as a Terminal or Modem program. Generally these are available commercially under various names; e.g. CROSS-TALK, MODEM-7, ASCII EXPRESS. Many such commercial programs have a capture function that allows uploaded files to be saved to a disk file. Frequently, a send file function, which allows a disk file to be downloaded to the SHOOTER, is also available.

2. COMPUTER OPERATING THE SHOOTER (File transfer) - A custom program may be written in either Basic or any other language that utilizes any or all of the commands supported by the SHOOTER. This allows the transfer of a HEX file to the SHOOTER or the reading of a HEX file from the SHOOTER (refer to Appendix A for details on hex formats). In this way, a turnkey operation may be created.

NOTE

The RAM buffer size on the SHOOTER is 128 Kbits, or 16 Kbytes.

3.0 SHOOTER COMMAND DICTIONARY

C	CHECKSUM	:	H	HEX LOAD
F	FILL RAM	:	G	HEX DUMP
N	BLANK CHECK	:	L	BINARY LOAD
P	PROGRAM	:	K	BINARY DUMP
R	READ EPROM	:	T	DOWNLOAD
V	VERIFY	:	U	UPLOAD
X	EXAMINE MEMORY	:	W	UPLOAD WITH WAIT
?	MEMU	:	I	OFFSET 00
I	INTEL HEX *			
M	MOTOROLA HEX			

Following are descriptions of the command character sequences sent via RS232 serial interface port.

NOTE

The identifier should not be sent. No carriage return should be sent after the command character(s). Lower case xx indicates a two digit hexadecimal number. All characters sent must be uppercase.

3.1 C = CHECKSUM ACD9

Directs the SHOOTER to calculate a checksum of the RAM data specified by the CNF plug. The resulting 16-bit sum is sent to the terminal or computer. The checksum is calculated by adding memory contents byte by byte and truncating the result to two bytes.

3.2 FXX = FILL RAM

Directs the SHOOTER to fill the entire RAM (as specified by the configurator plug) with any desired pattern. The function is followed by two hex digits representing the desired pattern.

3.3 G = HEXADECIMAL UP-LOAD

Directs the SHOOTER to transmit the contents of it's internal RAM to the terminal or computer. The amount that will be transmitted will be automatically determined by the configurator plug in the CNF socket.

For example: 2716 = 2K byte file
Data A2 (Hex) = 10100010 (binary)
2 Characters sent: A1(Hex) and 32(Hex)

3.4 H = HEXADECIMAL DOWN-LOAD

1. Directs the SHOOTER to enter the Hex DOWN-LOAD mode. All succeeding character values are stored beginning at location 0000.

2. If no character is received within approximately seven seconds, the unit will exit the DOWN-LOAD mode and return to the READY state. This is the normal way to exit this command.

For example: 2 Characters sent: A and 2
Data stored A2 (hex) = 10100010 (binary).

3.5 K = BINARY UP-LOAD

Directs the SHOOTER to transmit the contents of it's internal RAM to the terminal or computer. The data is sent as a character, byte by byte. The amount to be be transmitted will be determined automatically by the configurator plug in the CNF socket.

For example: 2716 = 2K
Data sent A2 (hex) = " (ASCII)
(Double quotes, MSB set)

3.6 L = BINARY DOWN-LOAD

1. Directs the SHOOTER to enter the DOWN-LOAD mode. The character's ASCII value is stored as data beginning at location 0000.

2. If no character is received within approximately seven seconds, the unit will exit the DOWN-LOAD mode and return to the READY state. This is the normal exit of this command.

For example: Data sent A and Z (ASCII)
Data stored location 0000 = 41,
Data stored location 0001 = 32.

3.7 N = BLANK CHECK

The BLANK CHECK command directs the SHOOTER to check an unknown EPROM for the blank data pattern. If the EPROM is not erased the SHOOTER will respond with the character sequence NB ERROR and light the ERROR indicator. If the target EPROM contains all FF's, the characters *EOJ* will be sent.

3.8 P = PROGRAM

Directs the SHOOTER to program the EPROM. A simple transfer takes place when the contents of the RAM are programmed into the coinciding locations of the EPROM.

NOTE

The EPROM being programmed must be blank or previously erased using a QUV-T8 EPROM eraser or equivalent.

1. Type a P on the terminal. The BUSY indicator will light.

2. After the EPROM has been completely programmed, all locations are then compared to the RAM. If any errors are found, (see VERIFY) the ERROR indicator will light. If you are using the Intellegent Programming Algorithm, the program cycle will stop at the first unprogrammable location.

3. If an error occurs, check the configurator plug for proper type and proper insertion in the CNF socket.

3.9 R = READ EPROM

The READ command directs the SHOOTER to read the EPROM into the SHOOTER internal RAM. Make sure that the EPROM is inserted correctly and the proper configurator plug is inserted in the CNF socket before issuing the READ command.

3.10 T = FORMATTED HEXADECIMAL DOWN-LOAD

1. Directs the SHOOTER to enter the DOWN-LOAD mode. The unit is now expecting the header character of the specified format, either an S for MOTOROLA or an : for INTEL. At this point all other characters will be ignored until the header is received. The most significant bit of an address will be ignored and any references to address space from 4000H to 7FFFH will not write to RAM. When the address is received, the contents of the offset register (see OFFSET command) will be subtracted from it and the result will be the actual RAM address where the data will be stored.

2. If no header character is received within approximately seven seconds, the unit will exit the DOWNLOAD mode and return to the READY state.

3. The checksum is compared at the end of each data record. If an error is detected, the ERROR indicator will light but the data will still be stored in RAM and the unit will still remain in this mode either until the end of the record is found or until transmission stops.

3.11 Uxx = FORMATTED HEXIDECIMAL UP-LOAD

1. Directs the SHOOTER to transmit the contents of it's internal RAM to the terminal or computer. The amount that will be transmitted will be automatically determined by the configurator plug in the CNF socket.

For example: 2716 = 2 Kbytes

2. The SHOOTER is now expecting two more characters. These characters will be two hexidecimal digits which are added to the SHOOTER addresses to form the address field in each record. Later, when the uploaded hex file is used on the host computer, it will be located at a valid memory location and not '0', which on many host computers is not usable as a data area.

For example: A 4 Kbyte Up-Load from a 2732

	Range of addresses in file	
UPLOAD WITH OFFSET COMMAND	STARTING ADDRESS	FINAL ADDRESS
U00	0000	0FFF
U30	3000	3FFF
U96	9600	A5FF
UCF	CG00	DEFF

3. The SHOOTER will transmit the contents of it's internal RAM buffer beginning at location 0000 in the hexi-decimal record format previously specified. If you do not change the format after power up, INTEL format will be used. A seven second delay before transmission is incorporated in order to allow setup time for the host computer.

3.12 V = VERIFY

This feature, whether directed from the terminal or the front panel switch, initiates a location-by-location comparison of the RAM buffer against the EPROM. If a discrepancy occurs, the red LED indicator will light. If all data compares, the green LED indicator lights. This is a very useful feature for comparing a large number of EPROMs against a known master.

3.13 W = UPLOAD WITH WAIT

This function is useful mainly when using the CPM PIP command or custom programs to UPLOAD to a host computer. The WAIT option causes a 25-second delay before transmission to allow the PIP command (or equivalent). In addition to a delay, the transfer is terminated with a control Z (1A hex), which is necessary to correctly close the file.

3.14 X = EXAMINE

The EXAM command allows you to gain direct access to a specific memory location in the SHOOTER internal RAM and, if desired, change the contents of that location.

1. To use this command type X.

2. A space will then be displayed.

3. The SHOOTER is now waiting for a 4-digit hex address. The specified address must be within the range of the EPROM being programmed.

4. After the address has been entered, the SHOOTER will display the address, an = sign, and the contents of this location.

5. If no change is to be made to this location enter a space (space bar) to increment the address and display the contents of the next location. To change the data, simply enter new data (two Hex digits).

6) To terminate the EXAM command, type a CR.

For example: Examine location 0200 and change location 0201 from 67 to 55 verify that location 201 contains 55.

User entry is in bold type.

```
* X 0200
0200=45 <SPACEBAR>
0201=67 55
0202=7A <CR>
*
* X 0201
0201=55 <CR>
*
```

3.15 ? = MENU

You may recall the SHOOTER command menu at any time by entering a question mark from the terminal.

3.16 I = INTEL HEXADECIMAL FORMAT

An I entered via the RS-232C port will direct all file transfer communications to be carried out in the INTEL hexadecimal format. See Appendix A for details and examples.

NOTE

The SHOOTER will default to INTEL upon power up.

3.17 M = MOTOROLA HEXADECIMAL FORMAT

An M entered via the RS-232C port will direct all communications to be carried out in the MOTOROLA hexadecimal format. See Appendix A for details and examples.

3.18 O = OFFSET

The OFFSET function is used in conjunction with the DOWNLOAD command and, once set, will retain its value until another OFFSET command is issued. When downloading formatted hex records the OFFSET value is subtracted from the most significant byte of each record.

For example:
```
Record start address.....0500
OFFSET value.............05
RAM address..............0000
```

NOTE

The OFFSET value defaults to 00 on power up.

4.Ø INDICATORS AND ERROR CODES

4.1 Status Indicators

There are three LED Status Indicators on the SHOOTER front panel. These Status Indicators are used in both stand-alone mode and RS-232 mode.

ERROR (Red LED Status Indicator)

Lights when the following faults occur:

A. An EPROM did not program correctly. Try erasing the EPROM again for a longer period of time. Also, check the configurator plug.

B. An error was found when performing a DOWNLOAD. Check baud-rate selection. Check for non-valid data in the host computer's hex file.

BUSY (Yellow LED Status Indicator)

Lights whenever the SHOOTER CPU is engaged in a command.

***** CAUTION *****

DO NOT remove or install an EPROM at any time when the BUSY (yellow status indicator) is lit or the device may be destroyed.

READY (Green LED Status Indicator)

Lights when the unit is in idle state. All power, address, and data lines have been taken low (i.e. less than 1 volt) making it safe to remove or install the EPROMs at this time.

4.1 ERROR CODES

Some commands will send an error message, indicating a failed condition, to the terminal. Following is a list of these errors and the coresponding error message.

```
BLANK CHECK.....................NB ERROR
PROGRAM (INTELIGENT ONLY).......PF ERROR
VERIFY..........................NV ERROR
```

After each command is completed, the SHOOTER will send the character sequence *EOJ* preceded by either a hex Ø6 (the code for ACK) if the function passed or a hex 15 (code for NAK) if it failed.

5.Ø DEVICE CONFIGURATION

In order to configure SHOOTER for a particular device type, you must plug one of the small DIP configurator plugs supplied with SHOOTER into the front panel CNF socket. These plugs are coded with a generic number representing the device type.

The configurator plug can be changed while power is on without damage to the unit but not while the BUSY (yellow status indicator) is lit. It is possible to edit the RAM buffer without the configurator plug inserted.

Data can be transferred from one EPROM device type to another by changing configurator plugs after the READ operation is complete. To avoid damage, do not insert any other type of hardware into the CNF socket. If additional or replacement Configurators are required, they can be obtained from LOGICAL DEVICES, INC.

NOTE

For some EPROMs the DIP switch located on the back of the unit (SWITCH #3) must also be set in addition to using the proper configurator. This switch controls the VCC voltage (5V or 6V) and the type of programming algorithm (standard or intelligent) used. For these devices, the DIP switch setting is indicated in the Device Dictionary under the column marked DIP SWITCH 3. ON (Down) is for intelligent programming and OFF (Up) is for standard programming. If you are using the 27256 EPROM, switch #4 will be used as the upper address bit (A14). The reason for this is that the 27256 EPROM is a 256 K-bit device and SHOOTER has only 128 K-bits of RAM buffer; therefore, programming must be done in two passes. The position of switch 4 is OFF (up) for the lower half (low bytes) of the 27256 and ON (down) for the upper half (high bytes) of the 27256.

To use the Device Dictionary simply locate the device type or its equivalent in the left column. Determine the number for the configurator plug you must insert in the CNF socket under the CNF column. Insert the EPROM in the ZIF socket.

6.Ø DEVICE DICTIONARY

NOTE

SHOOTER supports all of the devices listed in this Device Dictionary. Due to continual upgrading though, all supported devices may not be listed. For specific device requirements not listed, please call Logical Devices, Inc.

DEVICE SIZE	DEVICE TYPE	CNF #	STANDARD SUPPORT	SWITCH #3	SWITCH #4
AMD (Advanced Micro Devices)					
2K x 8 :	AM2716DC, AM2716nDC (all)	#2716	YES	OFF	OFF
4K x 8 :	AM2732	#2732	YES	OFF	OFF
4K x 8 :	AM2732A	#2732A	YES	OFF	OFF
8K x 8 :	AM2764	#2764	YES	ON	OFF
8K x 8 :	AM2764A	#2764A	YES	ON	OFF
16K x 8 :	AM27128	#27128	YES	ON	OFF
16K x 8 :	AM27128A	#27128A	YES	ON	OFF
	AM27256	#27256	YES	ON	OFF (low) ON (high)
Fairchild:					
4K x 8 :	F2732	#2732	YES	OFF	OFF
Fujitsu:					
2K x 8 :	MBM2716	#2716	YES	OFF	OFF
4K x 8 :	MBM2732	#2732	YES	OFF	OFF
4K x 8 :	MBM2732A	#2732A	YES	OFF	OFF
8K x 8 :	MBM2764	#2764	YES	ON	OFF
8K x 8 :	MBM27C64	#2764	YES	ON	OFF
16K x 8 :	MBM27128	#27128A	YES	ON	OFF
16K x 8 :	MBM27C128	#27128	YES	ON	OFF
32K x 8 :	MBM27C256 (21V)	#256-21	YES	ON	OFF (low) ON (high)
Hitachi:					
2K x 8 :	HN462716G	#2716	YES	OFF	OFF
4K x 8 :	HN462732G	#2732	YES	OFF	OFF
4K x 8 :	HN462732AG	#2732A	YES	OFF	OFF
8K x 8 :	HN482764G	#2764	YES	ON	OFF
16K x 8 :	HN27128	#27128	YES	ON	OFF
16K x 8 :	HN27C128	#27128	YES	ON	OFF
32K x 8 :	HN27256	#27256	YES	ON	OFF (low) ON (high)
32K x 8 :	HN27C256	#27256	YES	ON	OFF (low) ON (high)

DEVICE SIZE	DEVICE TYPE	CNF #	STANDARD SUPPORT	SWITCH #3	SWITCH #4
Intel Corp.:					
2K x 8 :	2716	#2716	YES	OFF	OFF
4K x 8 :	2732	#2732	YES	OFF	OFF
4K x 8 :	2732A	#2732A	YES	OFF	OFF
8K x 8 :	2764	#2764	YES	ON	OFF
8K x 8 :	2764A, 27C64	#2764A	YES	ON	OFF
16K x 8 :	27128	#27128	YES	ON	OFF
16K x 8 :	27128A	#128A	YES	ON	OFF
32K x 8 :	27256	#27256	YES	ON	OFF (low) ON (high)
Mitsubishi:					
2K x 8 :	M5L2716K	#2716	YES	OFF	OFF
4K x 8 :	M5L2732K	#2732	YES	OFF	OFF
4K x 8 :	M5L2732A	#2732A	YES	OFF	OFF
8K x 8 :	M5L2764K	#2764	YES	ON	OFF
16K x 8 :	M5L27128K	#27128	YES	ON	OFF
32K x 8 :	M5L27256K	#27256	YES	ON	OFF (low) ON (high)
Mostek:					
2K x 8 :	MK2716-n (all)	#2716	YES	OFF	OFF
Motorola:					
2K x 8 :	MCM2716, MCM27L16	#2716	YES	OFF	OFF
4K x 8 :	MCM2532	#2532	YES	OFF	OFF
4K x 8 :	MCM2732	#2732	YES	OFF	OFF
National Semiconductor:					
2K x 8 :	MM2716E, MM2716M, MMC2716M	#2716	YES	OFF	OFF
4K x 8 :	MMC27C32	#2732	YES	OFF	OFF
4K x 8 :	MM27C32A	#27C32A	YES	OFF	OFF
8K x 8 :	MM2764	#2764	YES	ON	OFF
8K x 8 :	MM27C64	#2764A	YES	ON	OFF
32K x 8 :	MM27C256 (21V)	#256-21	YES	ON	OFF (low) ON (high)
NEC					
32K x 8 :	27256	#256-21	YES	ON	OFF (low) ON (high)
32K x 8 :	27C256A	#27256	YES	ON	OFF (low) ON (high)

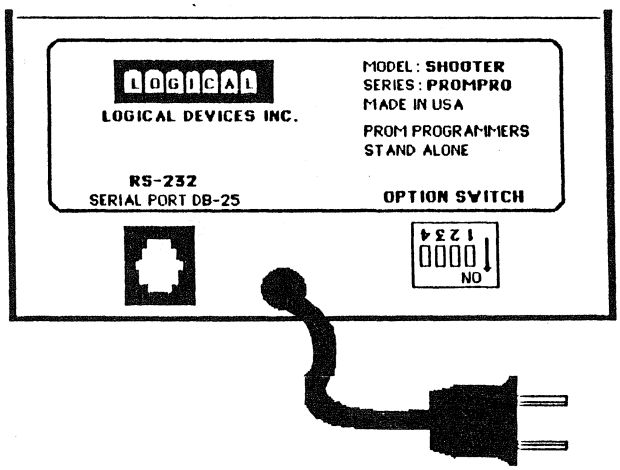
DEVICE SIZE	DEVICE TYPE	CNF #	STANDARD SUPPORT	SWITCH #3	SWITCH #4
SCS					
4K x 8 :	M2732A	#2732A	YES	OFF	OFF
Texas Instruments:					
2K x 8 :	TMS 2516	#2716	YES	OFF	OFF
4K x 8 :	TMS 2532	#2532	YES	OFF	OFF
4K x 8 :	TMS 2732	#2732	YES	OFF	OFF
4K x 8 :	TMS 2732A	#2732A	YES	OFF	OFF
8K x 8 :	TMS 2764	#2764	YES	ON	OFF
8K x 8 :	TMS 27C64	#2764A	YES	ON	OFF
16K x 8 :	TMS27128	#27128	YES	ON	OFF
16K x 8 :	TMS27C128	#27128A	YES	ON	OFF
16K x 8 :	TMS27128A	#27128A	YES	ON	OFF
32K x 8 :	TMS27C256	#27256	YES	ON	OFF (low) ON (high)
Toshiba					
4K x 8 :	2732	#2732	YES	OFF	OFF
4K x 8 :	2732A	#2732A	YES	OFF	OFF
8K x 8 :	2764	#2764	YES	ON	OFF
16K x 8 :	27128	#27128	YES	ON	OFF
32K x 8 :	27C256	#27256-21V	YES	ON	OFF (low) ON (high)
	2816	#2816	AR6 MODULE	OFF	OFF (low) ON (high)

7.0 COMMUNICATIONS

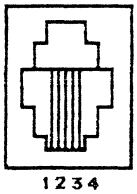
7.1 SERIAL INTERFACE

SHOOTER has a built-in serial I/O. The serial communication connector is located on the rear of the SHOOTER. See Appendix C for more information about the connector wiring.

The serial I/O can be directly connected to a CRT terminal (with pin 5 left disconnected). For most computer interfaces RS232 pins 2 and 3 must be reversed. It is always a good idea to refer to the pin-out of the serial interface for each terminal or computer to make sure that the wiring is correct.



SHOOTER RJ11



RS-232 Pin Assignment

1 = Tx data
2 = CTS (input to SHOOTER)
3 = Signal Ground
4 = Rx data (input to SHOOTER)

(View of RJ 11 socket from the back of SHOOTER)

Figure 7-1. SHOOTER RS-232C Pin Assignment

NOTE

SHOOTER provides an, internal pull-up resistor for the CTS line providing a user option to exercise this handshake line. Certain development systems may require long delays in responding to the serial port, in which case the CTS line to SHOOTER must be used. If CTS line is not activated by the computer or terminal, disconnect pin 5 from the RS232 connector cable to avoid accidently exercising this hand-shake line.

7.2 BAUD RATE SETTING

Prior to establishing communications between a terminal or computer and the SHOOTER, the baud rate switches on the SHOOTER rear panel must be set for the same baud rate as the terminal or computer. The baud rate switches are switches 1 and 2 of the four position DIP switch on the rear of the unit.

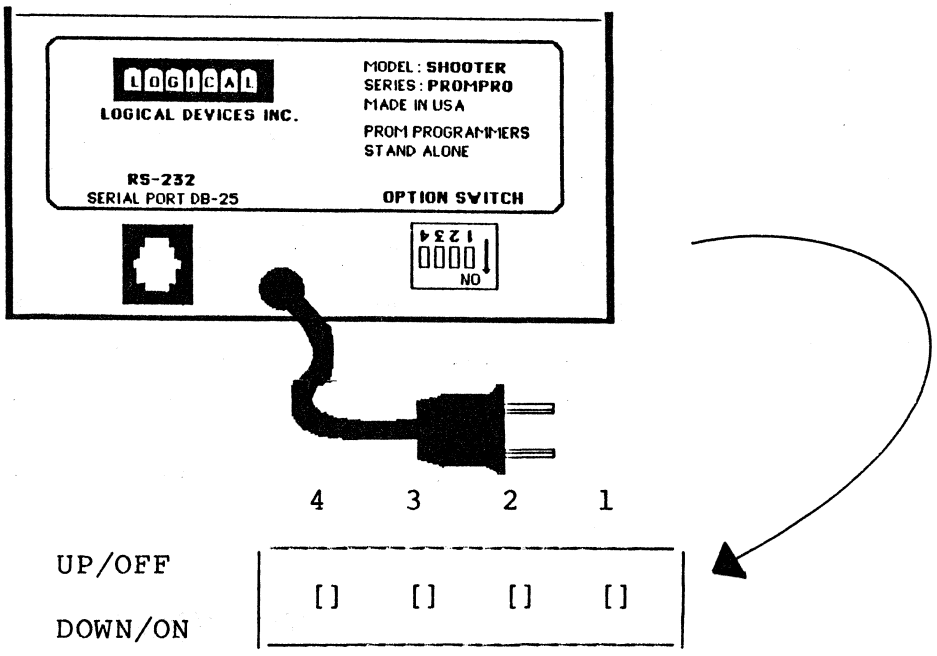


Figure 7-2. Baud Rate Setting Switches

BAUD RATE	SW 1	SW 2
110Ø	DN/ON	DN/ON
300Ø	UP/OFF	DN/ON
1200Ø	DN/ON	UP/OFF
2400Ø	UP/OFF	UP/OFF

SERIAL DATA FORMAT:

Half-Duplex
One start bit
8 data bits
Two stop bits
No parity

SERIAL SIGNAL LEVELS; +12V,-12V (CTS ENABLED = + 12V, NOT CTS = ON -12V)

8.0 SOFTWARE INTERFACE

In order to establish communication between SHOOTER and a computer, a computer RS-232C serial port at either 110, 300, 1200, or 2400 baud must be provided. In a typical development cycle the user assembles the program into an object file and then transfers that object file from either memory or disk to SHOOTER via the RS-232 interconnect. Some software development tools, such as assemblers and compilers, generate the assembled program output file in the form of INTEL or MOTOROLA hex formats. In such a case the job of sending the program information to the SHOOTER is only a matter of instructing the computer to transfer that output file to SHOOTER via the RS-232 interface.

Most operating systems provide a utility command to facilitate file transfer operations. For example, under the CPM operating system the PIP command is used to do such a task. Under PC DOS the COPY command is used. If your development system does not generate the standard formats, then a program to convert a binary or straight ASCII file into the standard HEX formats must be provided. The following software drivers are available, on floppy disk, from Logical Devices, Inc.:

IBM PC software driver	(Part No. SD-PC)
APPLE -II software driver	(Part No. SD-APPLE)
CPM 2.2 8" SS SD	(Part No. SD-CPM)
Commodore 64	(Part No. SD-C64)

It is very important to insure that the serial interface is working properly. If using the computer serial interface for the first time, there is always a possibility that it is not functioning properly. Read the instruction manual for your serial port very carefully. Monitor both the transmit and the receive line with an osciloscope to insure that data is being sent and received. In debugging any system always islolate each unknown and deal with one unknown at a time.

It is also necessary to understand how the serial communication for a particular computer is accomplished. Each system uses a slightly different hardware for the Serial/Modem port. For many systems a simple Terminal/Modem program is all that is required to operate the SHOOTER in the conversational mode. These programs are readily available for most systems.

A terminal program is basically a program that makes the computer look like a CRT terminal. All console entries are directed to the serial port and all data received from your serial port is

displayed on the screen. Prior to sending and receiving data to and from the serial port you must set the following parameters:

Start Bits	1
Stop Bits	2
No. Data Bits	8
Parity	Ignored
Baud	110, 300, 1200, 2400
Mode	Half Duplex

This format is software selectable in certain systems. For example, in a Radio Shack TRS-80 Model II with CPM 2.2 DOS, the SETUP command can be used to set parameters. If you do not have a terminal program and wish to write your own, there are three basic ways to write such programs.

1. Terminal program written in BASIC language.
2. Terminal program using the callable routines in Bios or System Monitor.
3. Assembly language program with your own Serial Port Drivers.

NOTE

All communication is done in ASCII Format. That is, when directing the programmer to program an EPROM, the ASCII equivalent of P (50H) is stored in the data register of the serial controller.

The I/O addresses in many systems are memory mapped. In other words, transferring a byte to and from the I/O port is similar to transferring a byte to and from a memory location. If your system is not I/O memory mapped, you must use the Input/Output instructions of your microprocessor with the corresponding I/O number assignment. Most serial controller chips have two internal registers, one Data and one Control/Status register. If you are writing directly to the serial controller you must always test the condition of the ready status bit to insure that you do not overrun previous data.

8.1 CREATING CUSTOM SOFTWARE INTERFACES

Software drivers consist of two distinct parts: a simple communications (terminal) mode and a Hex converter mode. The software drivers listed in Section 8.0 include either one or both parts. into them. If your system has either a terminal mode or a terminal program, a software driver may not be required. For example, the APPLE II software drivers do not have terminal mode because the APPLE II computer can function as a terminal by using a few simple control characters.

Appendix A lists several programs used to generate the Intel or Motorola Hex formats. In some instances the listing (6809 routines) is only part of a larger program. This partial listing is provided only as an illustration of how it is accomplished. An example of a TRS-80 color computer software driver is also provided. These routines can be used as a guide to develop your own program for any other computer system (refer to Appendix B).

NOTE

A file formatted as Intel or Motorola Hex is not required, but using formatted files allows error checking of records to be done.

Before attempting to analyze the program code it is a good idea to know what the general structure of the software program should look like. The following list defines the driver software structure (a slightly modified version of a standard TERMINAL program).

1. Initialize Serial port.
2. Scan Serial port for input.
3. If input, display on terminal.
4. Scan terminal for user entry.
5. If no entry, go to step 2.
6. Output user entry to serial port.
7. If user entry is T go to download routine.
8. If user entry is U go to upload routine, or go to step 2.

8.1.1 Download Routine

1. Prompt the user with the file name or memory addresses where the data is located.

2. Send either an I or an M command to set the programmer in the proper hex mode.

3. Send an ASCII T (54H) to the serial port to signal the programmer that the next entry is going to be a valid hex file.

4. Send Hex file to the serial port.

8.1.2 Upload Routine

1. Prompt the user for a disk file name or a memory address where the uploaded data is to be stored.

2. Send to the serial port an ASCII U followed by two hex digits indicating the page offset for the load address in system RAM.

3. Go to the Hex load routine.

4. Return.

NOTE

The driver routine address space must not coincide with the same address space as the upload file.

8.2 USING CP/M

Since CP/M is the most common operating system for microcomputers, the following are a few examples of its use with SHOOTER. For the sake of explanation it is assumed that a terminal program for your CP/M system has either been written or purchased for use. The name of this ficticious program will be MODEM.COM. Once power is applied to SHOOTER, the CP/M prompt A> should appear on the screen.

Next type MODEM and enter a carriage return <cr>. The screen will show:

```
A>MODEM
```

Once the program is activated it will provide prompts for several options. Select the Terminal option. If your SHOOTER is properly connected to the serial port and the serial port is operational, the menu should appear either when a power up procedure is performed or when a question mark is sent via the computer/terminal.

To set the programmer in the Intel hexadecimal mode, type I.

SHOOTER will respond with:

```
*EOJ*
*_
```

To examine and change a location in the SHOOTER RAM buffer, type an X, after the * prompt, followed by a four digit hex address of the location to be examined and/or changed. For this example use:

```
*X 0001
```

SHOOTER will respond with:

0001=hh

NOTE

The hh is a two digit hex value representing the contents of location 0001).

Either type new data or a space to skip:

```
0001=hh [SPACEBAR]
0002=hh [55] (as an example)
0003=hh [RETURN]  Return to prompt *
```

Now go back and examine location 0002. It should contain the data value 55.

Once the edit of the RAM buffer is complete, you can exit the terminal mode program and return to the CP/M operating system.

8.2.1 Using CP/M Submit To Download

Use the editor to create two permanent files. Each file should contain only one character - one file the I and the other file the T. Then using the SUBMIT facility of CP/M, send the files containing the I, the T, and your Hex file with only one command.

If you want to modify the format mode in the SHOOTER, create another permanent file containing either an I or an M. Then, using the SUBMIT facility, send this file as shown in the example below.

Example Submit File for downloading:

```
PIP PUN: = I.DWN
PIP PUN: = T.DWN
PIP PUN: = DATA.HEX
```

The file I.DWN contains only an I.
The file T.DWN contains only a T.
The file DATA.HEX contains data in Intel format to be transferred to the shooter RAM.

Many software development tools, such as assemblers or cross assemblers, have an option that allows the Hex address specification to be different than the actual assembly (ORIGIN) address. SHOOTER loads the Intel Hex file in its RAM buffer at the specified address of the Hex format. Since the programmer always programs the EPROM from location 0000, you must make sure that

the starting address of your file is directed to location 0000 of the SHOOTER RAM buffer. This is accomplished by creating another file that contains the character O (OFFSET COMMAND) followed by two hex digits equal to the PAGE number where your hex file will start. Next, simply send this file to the programmer before the hex file transfer.

Some 8080 Assemblers do no generate the proper Intel Hex Format End of File character. Instead, ten 00 characters are generated. If this last line of zeros is encountered, SHOOTER will generate an error. The easiest way to eliminate this condition is to delete the last line of the Hex File (ten 00's) using the Editor. SHOOTER will then simply time out 7 seconds after the file transfer is completed. Another method is to use the I parameter of the PIP command to ignore the ":00" records of Hex Format. The correct end of file record for Intel format is (:00000001FF). The correct end of file record for Motorola format is (S9030000FC).

8.2.2 Using CP/M PIP Command To Upload

The CP/M PIP command can also be used to UPLOAD a file to a host computer. However, instead of using the character U to initiate the UPLOAD function, the character W should be used (see UPLOAD with WAIT command). This causes a 25 second delay before transmission to allow time for the CP/M PIP command to execute. It will also terminate the transmission with a control Z, which is necessary for CP/M to close the file correctly.

Create a permanent file as described earlier containing only a U and name the file U.UPL. Create another submit file similar to the file described for downloading.

Example Submit File for uploading:

```
PIP PUN: = I.DWN
PIP PUN: = U.UPL
PIP PUN: = DATA.HEX
```

9.0 OPERATING TROUBLE GUIDE

If you have a problem when using SHOOTER, refer to the following Trouble Guide to determine whether the problem is either in operating the unit or a functional failure of the unit. The Trouble Guide is catagorized by general problem types. Each type is then listed by basic problem, the problem condition, and the possible remedy.

OPERATING TROUBLE GUIDE

General Problem Conditions:

UPON POWER-UP THERE IS NO LED INDICATION.

Assure that the line cord is inserted properly.

EPROM DOES NOT VERIFY AFTER READ.

If, after you have read the EPROM into the SHOOTER internal RAM buffer and depressed the VER switch, the red LED goes on, the following problems may exist:

Configurator plug inserted improperly or incorrect type. Remove the configurator plug and check for bent pins and proper internal connections (refer to CNF diagrams).

EPROM inserted improperly. Make sure that the EPROM pin 1 is in the proper position.

Defective EPROM, replace.

EPROM DOES NOT VERIFY AFTER PROGRAMMING

If, after the program cycle, the red LED goes on, it is an indication that the EPROM did not program properly. The following reasons could account for this program failure:

Check the configurator plug for proper type, bent pins, and proper insertion.

EPROM was not completely erased.

Defective EPROM, replace.

EPROM type is incorrect.

EPROM inserted backward.

Communications Problems:

MENU DOES NOT APPEAR ON THE SCREEN AFTER POWER UP

Chances are your cabling is incorrect. Refer to the cabling information in the serial interface section.

NOTE

In order to force the handshake lines to the proper levels, your terminal/computer may require that the following jumpers be installed on the terminal/computer side of the cable.

Jumper pin 4 to pin 5.
Jumper pins 6, 8, and 20.

MENU APPEARS ON SCREEN BUT UNIT HANGS-UP.

This is an indication that the SHOOTER serial handshake line CTS (pin 5) is at low signal level. This line is internally pulled up to +5V.

LED FLICKER ON DOWNLOAD

If you are downloading a file and you see the green and yellow LED flicker, the SHOOTER has not accepted the transfer (download) command. During a successful download operation the yellow (busy) LED is on continuously. This problem usually occurs because the user sends the T command immediately after the I or M. Since the SHOOTER responds with *EOJ* <cr> lf after it receives the I (INTEL Format) or M (Motorola format) command, it will miss any characters sent to it while it is busy responding. Attempt sending the I seperately in the terminal mode. After the T is received the programmer will ignore any character that is not a header character (S or :).

If no character is received within 7 seconds, the programmer will time-out. If you do not wish to send the I command manually in terminal mode, you may send it automatically:

Create a file with only the character I in the file and create another file with only the character T. Now use the command chain capability of your operating system to send the above files and the Hex file to the programmer with one simple command.

NOTE

This procedure applies to most systems with the exception that certain command names and system parameters may vary. In all cases, the important fact to remember is, if the computer sends any character other than the

SHOOTER commands (with the exception of nulls), the SHOOTER will respond in half duplex with an *. Suppose, when sending the character I, that the computer actually precedes it with a control character. Then SHOOTER is going to miss the I because it is busy sending the *. To avoid this, allow several character times for response. If you encounter error problems during transmission, check the ground connection in your cable. Also, make sure your system is configured for:

ONE START BIT
8 DATA BITS
2 STOP BITS
NO PARITY

10.0 PERIODIC CALIBRATION CHECK

SHOOTER is calibrated and tested for proper programming voltages at the factory, prior to shipment. Typically, additional calibration is not required. However, since even a small program voltage change can result in damage to an EPROM, it is recommended that periodic program voltage calibration checks be made.

NOTE

Program voltage changes only occur when the configurator plug or an IC is not inserted properly or in the instance of a limited number of certain component failures.

The following equipment is required for the programming voltage calibration check:

Digital Voltmeter	3 1/2 digit
Small clip leads	2
Test pins	2
Resistor	1K, 1 watt

Voltage checks are performed on the ZIF socket using a 1K, 1 watt resister in the proper ZIF socket pin locations. The DVM clip leads are then attached to this resistor for VPP voltage measurements. One end of the resister is always used for the ground connection (pin 14 on the ZIF socket). The other end is inserted into the corresponding program voltage socket pin.

10.1 25 VOLT PROGRAM VOLTAGE CHECK

The following procedure verifies the calibration voltage for all EPROM's requiring a 25 volt programming voltage.

1. Insert configurator plug #2716 in the CNF socket.

2. Insert a 1K resister into the ZIF socket between pins 23 and 14 (close the ZIF socket handle for firm grip).

3. Connect the DVM ground lead to pin 14 of the ZIF socket. Connect the positive lead to pin 23 of the ZIF socket.

4. Depress the RESET button to reset SHOOTER. When the BUSY (yellow LED) indicator goes out, the DVM should read less than +2 VDC.

5. Depress the PROG button. The BUSY indicator should light and the DVM should read between +24.5 and +25.5 VDC.

10.2 21 VOLT PROGRAM VOLTAGE CHECK

The following procedure verifies the calibration voltage for all EPROM's requiring a 21 volt programming voltage (VPP) such as the 2732A EPROM.

1. Insert configurator plug #2732A in CNF socket.

2. Insert test pins in ZIF socket pins 14 and 22.

3. Connect the DVM ground lead to pin 14 of the ZIF socket. Connect the positive lead to pin 22 of the ZIF socket.

4. Depress the RESET button to reset SHOOTER. When the BUSY (yellow LED) indicator goes out, the DVM should read less than +2 VDC.

5. Depress the PROG button. The BUSY indicator should light and the DVM should read between +20.5 and +21.5 VDC.

APPENDIX A

HEX FORMATS

INTEL LOAD MODULE FORMAT

FORMAT DESIGNATION	No. of ASCII CHARS.	CONTENT DESCRIPTION
Header Character	1	Always a colon "1" (HEX 3A)
Record Length	2	A two-digtit HEX value specifiying the number of data bytes in the record. A record length of Ø indicates the last record in the file.
Starting Address	4	Four HEX digits representing the address of the memory location where the first Data Byte will be located.
Type of Record	2	ØØ-Normal data record Ø1-End of file record
Checksum	2	Two-digit HEX number representing two's complement of the sum, modulo 256 of the preceding eight bit bytes.

EXAMPLE:

:1ØØØØØØØ27Ø4ØDA8189327D75BF8CØ43A9A8FA83F

: 1Ø ØØØØ ØØ 27 Ø4 ØD A5 15 93 27 D7 D5 BF 8C Ø4 3A 9A 8F A8 3E

Type of Record
Starting Address
Record Length
Header
Data
Checksum

AN EXAMPLE OF A 1K BYTE INTEL OBJECT MODULE:

```
:1000000027040DA5159327D7D5BF8C043A9A8FA83E
:10001000A999CF54170A7221347902346B961804C7
:1000200035994F14F39ACF088A209A7F0214EF363B
:1000300032E7D99625D5ABBF43B53435144234280C1
:10004000043C65C589DF9AA08A802702A8A9AA8AEC
:100050009023C03A8654050937928B6850AD2629534
:1000600004B67657455D5535FAF99CF9ABF03B3B3
:100070009674AB930304967BBB029303F5C59682FB
:10080006406079 6A9C5AE54CE47AABC60541BECC3
```

```
:10090008D54708A080A729EBE4026A004A226A231
:1000A000BE80D5FBC532CD24E607C65F03FE96B2FF
:1000B00064BF030896069A8F99CF54178A080A726C
:1000C000C3243BB6CB26C92487442824062033A54AC
:1000D000872310AB54A52754785481175412EECD92
:1000E000233A54872754A52301547823EF4412C59B
:1000F00018F896F519C58AA09AEFF802F943B03AB4
:100100009AEF80230893894854179977F541727C57D
:10011000A98A0814F59ACF08AF8A209A7F54FF1451
:10012000EF3624E7D9961593BF8509B2312306240B
:1001300033231554891FFFE3963393948FF8B65FE4
:1001400002F93A43103A3A38002997F238F3923D1
:100150005C541D8940997F8980346B8A20963BA826
:10160000A93479AF54FF346B9661938A8018F8965E
:1001700078192304367 7E7D983F802F943803A3AAD
:100180008AB080089A4F83C514F59A6F800299B798
:100190008A88541B9AF718F8968819F972062488E9
:1001A000BAC8D0DBBCCAD5E0890609530677703A0DC
:1001B000B3890609530677703A4B334BCD5BCF0EC6D
:1001C000BFECC1BC1BECC59334CAD5BCBCECCD9311
:1001D000D5BC29ECD3D5BC2AECD893D5BC11ECDE28
:1001E000D5BC10ECE39323535487BB002331540850
:1001F000EEE623535487233954872303547827543 6
:10020000782754782 3FC44125487231354FA54A5B6
:100210054B154785470935419541D238362551665
:10022006021542316274423838AA09AEFF802F94326
:10023000303A800299B79ADF541B8A2018F8962822
:10024000 19F93792282406EC53040686 4BD505BCD1
:10025000DC16477651D5B515BA01B90034B19734DB
:10026000A8FAF72A49F66C866AA9445F34B1F99373
:10027000230D5489230A4489C5AD47547EFD530F8D
:1002800003305717D287075687D5A8BA019799FE2A
:10029000 0834A8F69FFAF72A58C68E890144918936
:1002A0000134A824A85478C5F96A54ADF854FA4426
:1002B0007814F5BF109A6F0854FA547814EFEFB51C
:1002C000FB3793544BC554D047AE54CE4E93544B4A
:1002D00003D0F2E3975792DBB2DB9303EFF2E30331
:1002E000FAF2E68920930310 3C554EF530FA954F3
:1002F000C354FAA89354C39A6F02C52B6B2B93086F
:10030000DFC605892093D555FB3291544BD35396C4
:100310000B85544B530FC5C60BBB00D503F7962175
:10032009554C354FA03FDAD54E9D5FDC63814F510
:1003300643414EF54F5ED3254C3C537DBD5740182
:100340095B60B93111A0D0A6528582961 6D0D0A8F
:10035000496E74656C0D0A4D6F746F726F6C610D30
:100360000A50726F6772616D0D0A5472616E736626
:10037000 65720D0A55706C6F61640D0A566572697D
:10038000 66790D0A2A000D0A2A454F4A2A0D0A2AC3
:10039000 0544BD33A96918554C3ADC5AB54E9D5BF
:1003A00054C354FAC6AA95FDC6B414F564B014EF4C
:1003B00054F5EDAE54C3C56BD5740195B691932336
:1003C00020548754E9D55470C5230F59A9D554A793
:1003D000233D546714F59A6F0854782320548754 8A
:1003E0004B03E0C6FC0313C6BEF999DF54C5AC0944
:1003F000B2C6FC9008DCC6FC233F548714EF64C6E9
:00000001FF
```

Sample Intel hexadecimal record :

:080AB8001122334455667788D2

Description of sample Intel Record

:	record header charcter
08	number of data bytes (in hexadecimal)
0AB8	address where first data byte will be loaded (not including any offset specified)
00	record type - 00 is data, 01 is end of file (types 02 and 03 not supported by PromPro 8)
11	data byte
22	.
33	.
44	.
55	.
66	.
77	.
88	data byte
D2	checksum value

Calculating the Intel record checksum

```
        08
        0A
        B8
        00
        11                              100
Step 1  22                 Step 2    -  2E
        33
        44                              D2 = 2's complement of 2E
        55
        66            (Intel checksum value for this record)
        77
        88

        2E  = sum modulo 256
```

MOTOROLA LOAD MODULE FORMAT

FORMAT DESIGNATION	NO. of ASCII CHARS.	CONTENT DESCRIPTION
Start of Record	1	Always an "S" (HEX53)
Type of Record	1	Ø-Header record (sometimes not used) 1- Normal data record 9-End of file record
Byte Count	2	A two-digit HEX value specifying the number of data bytes in the record, including the address and checksum. Each byte is represented as two HEX characters.
Address	4	Four HEX digits representing the address of the memory location where the first Data Byte will be located.
Checksum	2	Two-digit HEX number representing one`s complement of the sum, modulo 256, of the Byte Count, Address, and the Data Byte.

EXAMPLE :

S1130000270400DA5159327D7D5BF8C043A9A8FA83A (Normal data record)

S 1 13 0000 27 04 0D A5 15 93 27 D7 D5 BF 8C 04 3A 9A 8F A8 3A

- S — Start of Record (Header character)
- 1 — Type of Record
- 13 — Byte Count
- 0000 — Starting Address
- 27 04 0D A5 15 93 27 D7 D5 BF 8C 04 — Data
- 3A 9A 8F A8 3A — Checksum

EXAMPLE OF A 1K BYTE MOTOROLA OBJECT MODULE:

```
S113000027040DA5159327D7D5BF8C043A9A8FA83A
S1130010A999CF54170A7221347902346B961804C3
S113002035994F14F59ACF088A209A7F0214EF3637
S113003032E7D99625D5ABBF43B5343514423428BD
S1130040043C65C589DF9AA08A802702A8A9AA8AE8
S11300509023C03A865405093792B6850AD2629530
S1130060048676575455D5535FAF99CF9ABF03B3AF
S11300709674AB930304967BBB029303F5C59682F7
S1130080640607 96A9C5AE54CE47AABC60541BECBF
S11300908D54708A080A729EBE4026A004A226A22D
```

```
S11300A0BE80D5FBC532CD24E607C65F03FE96B2FB
S11300B064BF030896069A8F99CF54178A080A7268
S11300C0C3243BB6CB26C92487442824062 33A54A8
S11300D0872310AB54A527547854B1175412EECD8E
S11300E0233A54872754A523015478 23EF4412C597
S11300F018F896F519C58AA09AEFF802F943B03AB0
S11301009AEF80230893894854 17997F541727C579
S1130110A98A0814F59ACF08AF8A209A7F54FF144D
S1130120EF3624E7D9961593BF8509B23123062407
S1130130323231554891FFFE39633939A8FF8B65FE0
S113014002F93A43103A3A3A8002997F238F3923CD
S11301505C541D8940997F8980346B8A20963BA822
S1130160A93479AF54FF346B9661938A8018F8965A
S1130170781923043677E7D983F802F943803A3AA9
S11301808AB080089A4F83C514F59A6F800299B794
S11301908A88541B9AF718F8968819F972062488E5
S11301A0BAC8D0DBBCCAD5E08906095306 7703A0D8
S11301B0B38906095306 7703A4B334BCD5BCF0EC69
S11301C0BFECC1BC1BECC59334CAD5BCBCECCD930D
S11301D0D5BC29ECD3D5BC2AECD893D5BC11ECDE24
S11301E0D5BC10ECE39323535487BB00233154084C
S11301F0EEE6235354872339548723035478275432
S1130200782754782 3FC44125487231354FA54A5B2
S1130210 54B154785470935419541D238362551661
S1130220215423162744 23838AA09AEFF802F94322
S1130230303A800299B79ADF541B8A2018F896281E
S1130240 19F93792282406EC530406864BD505BCCD
S1130250DC16477651D5B515BA01B90034B19734D7
S1130260A8FAF72A49F66C866AA9445F34B1F9936F
S1130270230D5489230A4489C5AD47547EFD530F89
S1130280033057 17D287075687D5A8BA019799FE26
S1130290 0834A8F69FFAF72A58C68E890144918932
S11302A00134A824A85478C5F96A54ADF854FA4422
S11302B07814F5BF109A6F0854FA547814EFEFB518
S11302C0FB3793544BC554D047AE54CE4E93544B46
S11302D003D0F2E3975792DBB2DB9303EFF2E3032D
S11302E0FAF2E689209303 093C554EF530FA954EF
S11302F0C354FAA89354C39A6F02C52B6B2B93086B
S1130300DFC605892093D555FB3291544BD35396C0
S1130310 0B85544B530FC5C60BBB00D503F7962171
S1130320 9554C354FA03FDAD54E9D5FDC63814F50C
S1130330643414EF54F5ED3254C3C537DBD574017E
S1130340 95B60B93111A0D0A6528582961 6D0D0A8B
S1130350496E74656C0D0A4D6F746F726F6C610D2C
S1130360 A50726F6772616D0D0A5472616E736622
S1130370 5720D0A55706C6F61640D0A5665726979
S1130380 6790D0A2A000D0A2A454F4A2A0D0A2ABF
S1130390 0544BD33A96918554C3ADC5AB54E9D5BB
S11303A054C354FAC6AA95FDC6B414F564B014EF48
S11303B054F5EDAE54C3C56BD5740195B691932332
S11303C0 20548754E9D55470C5230F59A9D554A78F
S11303D0233D548714F59A6F0854782320 54875486
S11303E04B03E0C6FC0313C6BEF999DF54C5AC0940
S11303F0B2C6FC9008DCC6FC233F548714EF64C6E5
S9030000FC
```

S9030000FC (end of file record)

Sample Motorola hexadecimal record:

S1ØB0AB81122334455667788CE

Description of sample Motorola record

S	record header character
1	record type - 1 is data record, 9 is end of file record (type 2 and type 3 not supported)
ØB	number of bytes to follow in record, including checksum byte
ØAB8	address where first data byte will be loaded (not including and offset specified)
11	data byte
22	.
33	.
44	.
55	.
66	.
77	.
88	data byte
CE	checksum

Calculating the Motorola record checksum

```
        ØB
        ØA
        B8
        ØØ
        11                  31(hex) ---> ØØ11 ØØØ1 (binary)
        22                                       |
Step 1  33          Step 2                       |
        44                                       |
        55                  CE(hex) <--- 11ØØ 111Ø (binary)
        66
        77              (Motorola checksum value for this record)
        88

        31  = sum modulo 256
```

Sample Binary download (unformatted):

Character Sent	ASCII Value (decimal)	Order of Sending	PROMPRO-8 RAM location where stored	Data Stored (hex)	Data Stored (binary)
1	49	1st	ØØØØ	31	ØØ11 ØØØ1
2	5Ø	2nd	ØØØ1	32	ØØ11 ØØ1Ø
3	51	3rd	ØØØ2	33	ØØ11 ØØ11
A	65	4th	ØØØ3	41	Ø1ØØ ØØØ1
a	193	5th	ØØØ4	A2	1Ø1Ø ØØ1Ø
carriage ret	13	6th	ØØØ5	ØD	ØØØØ 11Ø1
?	47	7th	ØØØ6	2F	ØØ1Ø 111[illegible]

8080 routine

TO GENERATE INTEL HEX FILE

```
        TITLE   'UNLOAD ver 2.1 - Create HEX file from COM file'
;       (revised 05/20/81)
;
;--->Needs MAC and SEQIO.LIB to assemble<---
;
;Originally from CPMUG 29.23
;
;05/20/81 Modified for 32 bytes/record instead of 16, for less
;         overhead in the .HEX output file, by Dav Holle.
;
;11/07/80 Modified to default to 100H, increase size of buffers,
;         add signon message.  By Keith Petersen, W8SDZ
;
;To use, type: UNLOAD <FILENAME> <ADDR>
;
;Where: <FILENAME>.COM is the source file
;       <FILENAME>.HEX will be the output file
;       <ADDR> is the optional start address in hex (default=100)
;
        ORG     100H

        MACLIB  SEQIO   ;DEFINE MACRO LIBRARY USED
;
        LHLD    6       ;GET BASE OF BDOS
        DCX     H       ;BACK OFF ONE BYTE
        SPHL            ;SET STACK THERE
        CALL    SIGNON
        DB      CR,LF,'UNLOAD ver 2.1',CR,LF,'$'
        DB      '05/20/81' ;REVISION DATE (doesn't print)
;
SIGNON: POP     D       ;GET MSG ADR
        MVI     C,@MSG  ;PRINT IT
        CALL    @BDOS
        LXI     H,100H  ;DEFAULT UNLOAD ADRS
        LXI     D,FCB2+1
        LDAX    D       ;GET OPTION ADRS
        CPI     ' '     ;ANY GIVEN?
        JZ      INITFL  ;NO, DEFAULT TO 100H
        LXI     H,0
        MVI     B,0
;
ADRLUP: LDAX    D
        INX     D
        SUI     '0'
        JC      INITFL
        CPI     10
        JC      ADDNIB
        SUI     7
        JC      INITFL
        CPI     16
        JNC     INITFL
;
ADDNIB: DAD     H
```

```
          DAD     H
          DAD     H
          DAD     H
          MOV     C,A
          DAD     B
          JMP     ADRLUP
;
INITFL:   PUSH    H
          FILE    INFILE.SOURCE,,1,COM,2048
          FILE    OUTFILE,OUTPUT,,1,HEX,2048
          POP     H
;
ADRDON:   SHLD    LODADR
;
UNLOOP:   GET     SOURCE
          JZ      GEOF
          PUSH    PSW
          MVI     A,':'
          PUT     OUTPUT
          XRA     A
          STA     CHEKS
          MVI     A,32      ;was 16
          CALL    PUTBYTE
          LDA     LODADR+1
          CALL    PUTBYTE
          LDA     LODADR
          CALL    PUTBYTE
          XRA     A
          CALL    PUTBYTE
          POP     PSW
          MVI     B,32      ;was 16
;
LINLUP:   PUSH    B
          CALL    PUTBYTE
          POP     B
          DCR     B
          JZ      NEXTL
          GET     SOURCE
          JMP     LINLUP
;
PUTBYTE:  MOV     C,A
          LDA     CHEKS
          SUB     C
          STA     CHEKS
          MOV     A,C
          RRC
          RRC
          RRC
          RRC
          CALL    PUTNIB
          MOV     A,C
;
PUTNIB:   ANI     0FH
          ADI     '0'
          CPI     '9'+1
          JC      PUTNB1
          ADI     7
;
PUTNB1:   PUSH    B
          PUT     OUTPUT
          POP     B
          RET
;
NEXTL:    LDA     CHEKS
          CALL    PUTBYTE
          MVI     A,CR
```

```
          PUT       OUTPUT
          MVI       A,LF
          PUT       OUTPUT
          LHLD      LODADR
          LXI       D,32      ;was 16
          DAD       D
          JMP       ADRDON
;
GEOF:     MVI       A,':'
          PUT       OUTPUT
          MVI       B,5
;
GEOF1:    XRA       A
          PUSH      B
          CALL      PUTBYTE
          POP       B
          DCR       B
          JNZ       GEOF1
          MVI       A,CR
          PUT       OUTPUT
          MVI       A,LF
          PUT       OUTPUT
          FINIS     OUTPUT
          LXI       D,DMSG
          MVI       C,@MSG
          CALL      @BDOS
          JMP       0
;
DMSG:     DB        'DONE',CR,LF,'$'
;
LODADR:   DS        2
CHEKS:    DS        1
FCB2      EQU       6CH
;
BUFFERS   EQU       $         ;INPUT/OUTPUT BUFFERS GO HERE
;
          END
```

8080 Program to Generate Intel Hex Format

```
        READ COMMAND   READ INTEL HEX FORMAT TAPE (W/BIAS IF DESIRED)
READ:   CALL    EXPRI  ;GET BIAS INPUT IF DESIRED
        POP       H    ;PLACE BIAS INTO HL (IS 0000 OF NO BIAS)
REDO:   CALL      RI   ;READ FROM LOGICAL TAPE READER DEVICE
        RC             ;RETURN IF NO BYTE IS AVAILABLE
        ANI      7FH   ;MASK OFF POSSIBLE BITS
        SUJ      ':'   ;CLUMSY COMPARISON FOR START LINE
        JNZ      REDO  ;LOOP TIL START OF LINE DELIMINATOR FOUND
        MOV       D,A  ;A MUST BE ZERO, SO USE IT TO CLEAR D RFG
        PUSH      H    ;SAVE BIAS VALUE ON STACK
        CALL    BYTE   ;GET FIRST DATA BYTE
        JZ      RED2   ;IF ZERO, INDICATES LAST LINE AND LOAD ADDRESS
        MOV       F,A  ;LOAD NUMBER OF DATA BYTES IN LINE INDICATOR IN F
        CALL    BYTE   ;GET FIRST ADDRESS BYTE
        MOV       B,A  ;SAVE HIGH ORDER ADDRESS IN B
        CALL    BYTE   ;GET LOWER ADDRESS BYTE
        MOV       C,A  ;SAVE LOW ORDER ADDRESS IN C
        DAD       B    ;ADD BIAS TO LOADING ADDRESS FROM LINE
        CALL    BYTE   ;JUST READ TERMINATOR BYTE
RED1:   CALL    BYTE   ;READ DATA BYTE
        MOV       M,A  ;STORE DATA BYTE AT LOAD ADDR+BIAS
        INX       H    ;BUMP STORAGE POINTER
        DCR       F    ;DECREMENT BYTE COUNT FOR LINE
        JNZ     RED1   ;LOOP TILL BYTE COUNT IS EXHAUSTED
        CALL    BYTE   ;READ CHECK BYTE AND DO FINAL SUM
        POP       H    ;RESTORE BIAS FOR NEXT LINE
        JZ      REDO   ;JUMP FOR NEXT LINE IF CHECKSUM=0
        STC            ;SET THE CARRY BIT ON AS AN ERROR INDICATOR
        RFT
RED2:   CALL    BYTE   ;GET TWO ADDRESS BYTES AND BRANCH THERE
        MOV       H,A
        CALL    BYTE
        POP       B    ;REMOVE BIAS VALUE FROM STACK
        MOV       L,A  ;CHECK THE EXECUTION ADDRESS FOR ZERO
        URA       H
        RZ             ;RETURN TO MONITOR ON A ZERO EXECUTION ADDR
        PCHl           ;GO TO EXECUTION ADDRESS
BYTE:   CALL   RNBBL   ;READ UPPER NIBBLE OF DATA BYTE
        RLC            ;SHIFT TO UPPER 4 BITS
        RLC
        RLC
        RLC
        MOV       C,A  ;SAVE MS NIBBLE IN C
        CALL   RNBBL   ;READ SECOND NIBBLE
        ORA       C    ;COMBINE NIBBLES TO FORM HEX BYTE
        MOV       C,A  ;SAVE BYTE IN C
        ADD       D    ;ADD TO CHECK SUM
        MOV       D,A  ;SAVE NEW CHECK SUM
        MOV       A,C  ;RESTORE DATA BYTE TO ACCUM
```

```
RNBBl:    CALL      RJ    ;DO READER INPUT
          JC      RNBER   ;IF CARRY SET, ERROR
          ANJ       ZFH   ;MASK OFF PARITY BIT
          CALL    NIBBL   ;READ NIBBLE FROM ASCII HEX BYTE
          JC       RNBR   ;JUMP ON ERROR
          RET
RNBFR:    POP         H    ;RETURN TO MONITOR ON ERROR
          POP H
          POP H
          RET
```

6809 SAMPLE PROGRAM TO GENERATE MOTOROLA HEX FORMAT

```
FC11 17   01A1  LOAD1   LBSR    ECHON    INBUT 8 BIT BYTE WITH NO ECHO
FC14 81   53    LOAD2   CMPA    #'S      IS IT S, START CHAR?
FC16 26   F9            BNE     LOAD1    NO, DISCARD AND GET NEXT CHAR
FC18 17   019A          LBSR    ECHON
FC1B 81   39            CMPA    #'9      IS 9, END OF FILE CHAR?
FC1D 27   3D            BEQ     LOAD21   IS SO, EXIT LOAD
FC1F 81   31            CMPA    #'1      IS 1, FILE LOAD CHAR?
FC21 26   F1            BNE     LOAD2    IF NOT, LOOK FOR START CHAR
FC23 17   0116          LBSR    BYTE     INPUT BYTE CNT
FC26 34   02            PSHS    A        PUSH COUNT ON STK
FC28 29   26            BVS     LODERR   (V)C-CODE SET, ILLEGAL HEX
FC2A 17   00FE          LBSR    IN1ADR   INPUT LOAD ADDR
FC2D 29   21            BVS     LODERR   (V) C-CODE SET, ADDR NOT HEX
FC2F 34   10            PSHS    X        PUSH ADDR ON STK
FC31 E6   E0            LDB     ,S+      LOAD MSB OF ADDR AS CKSUM BYTE
FC33 EB   E0            ADDB    ,S+      ADD LSB OF ADDR TO CKSUM
FC35 EB   E4            ADDB    ,S       ADD BYTE COUNT BYTE TO CKSUM
FC37 6A   E4            DEC     ,S       $FC37 DEC BYTE COUNT 2 TO BYPASS
FC39 6A   E4            DEC     ,S       ADDRESS BYTES
FC3B 34   04    LOAD10  PSHS    B        PUSH CKSUM ON STK
FC3D 17   00FC          LBSR    BYTE     INPUT DATA BYTE - 2 HEX CHARS
FC40 35   04            PULS    B        POP CKSUM FROM STK
FC42 29   0C            BVS     LODERR   (V) SET, DATA BYTE NOT HEX
FC44 34   02            PSHS    A        PUSH DATA BYTE ON STK
FC46 EB   E0            ADDB    ,S+      ADD DATA TO CKSUM, AUTO INC STK
FC48 6A   E4            DEC     ,S       DEC BYTE COUNT 1
FC4A 27   05            BEQ     LOAD16   IF BYTE COUNT ZERO, CHECK CKSUM
FC4C A7   80            STA     ,X+      SAVE DATA BYTE IN MEM
FC4E 20   EB            BRA     LOAD10   GET NEXT DATA BYTE
FC50 SF           LODERR  CLRB             ERR COND, ZERO CKSUM
FC51 35   02    LOAD16  PULS    A        ADJUST STK (REMOVE BYTE CNT)
FC53 C1   FF            CMPB    #$FF     CKSUM OK?
FC55 27   B2            BEQ     LOAD     IF SO, LOAD NEXT LINE
FC57 86   3F            RDA     #'?      LOAD "?", ERR INDICATOR
FC59 17   0183          LBSR    OUTCH    OUTPUT TO TERM
FC5C 73   DFE2  LOAD21  COM     ECHO     TURN ECHO ON
FC5F 86   13            LDA     #$13     $FC5F LOAD "DC3" CASS READ OFF COD
FC61 16   017B          LBRA    OUTCH    OUTPUT IT
                        *
                        * "P" PUNCH MIKBUG TAPE
                        *
FC64 6F   E2    PUNCH   CLR     ,-S      CLEAR RES BYTE ON STK
FC66 17   00B7          LBSR    IN2ADR   GET BEG & END ADDR
FC69 34   30            PSHS    X,Y      SAVE ADDR'S ON STK
FC6B 29   4A            BVS     PUNEXT   (V) C-CODE SET, EXIT PUNCH
FC6D AC   62            CMPX    2,S      CMP BEG TO END ADDR
FC6F 25   46            BCS     PUNEXT   IF BEG>END, EXIT PUNCH
FC71 30   01            LEAX    1,X      INC END ADDR
FC73 AF   E4            STX     ,S       STORE END ADDR ON STK
FC75 86   12            LDA     #$12     LOAD 'DC2' PUNCH ON CODE
```

```
FC77 17   0165          LBSR   OUTCH    OUTPUT TO TERM
FC7A EC   E4    PUNCH2  LDD    ,S       LOAD END ADDR IN D REG
FC7C A3   62            SUBD   2,S      SUB BEG FROM END
FC7E 27   06            BEQ    PUNCH3   SAME, PUNCH 32 BYTES OF DEFAULT
FC80 1083 0020          CMPD   #$20     LESS THAN 32 BYTES?
FC84 23   02            BLS    PUNCH4   PUNCH THAT MANY BYTES
FC86 C6   20    PUNCH3  LDB    #$20     LOAD BYTE CNT OF 32
FC88 E7   64    PUNCH4  STB    4,S      STORE ON STK AS BYTE CNT
FC8A 8E   FEDC          LDX    #MSG20   POINT TO MSG "S1"
FC8D 17   010E          LSBR   PSTRNG   PRINT MSG
FC90 CB   03            ADDB   #3       ADD 3 BYTES TO BYTE CNT
FC92 1F   98            TFR    B,A      GET BYTE CNT I A-REG TO PUNCH
FC94 17   00DB          LBSR   OUT2H    OUTPUT BYTE COUNT
FC97 AE   62            LDX    2,S      LOAD BEG ADDR
FC99 17   00CE          LBSR   OUT4H    PUNCH ADDR
FC9C EB   62            ADDB   2,S      ADD ADDR MSB TO CKSUM
FC9E EB   63            ADDB   3,S      ADD ADDR LSB TO CKSUM
FCA0 EB   84    PUNCHL  ADDB   ,X       ADD DATA BYTE TO CKSUM
FCA2 A6   80            LDA    ,X+      LOAD DATA BYTE TO PUNCH
FCA4 17   00CB          LBSR   OUT2H    OUTPUT DATA BYTE
FCA7 6A   64            DEC    4,S      DEC BYTE CNT
FCA9 26   F5            BNE    PUNCHL   NOT DONE, PUNCH NEXT BYTE
FCAB 53                 COMB            1'S COMPLIMENT CKSUM BYTE
FCAC 1F   98            TFR    B,A      PUT IN A REG TO PUNCH
FCAE 17   00C1          LBSR   OUT2H    OUTPUT CKSUM BYTE
FCB1 AF   62            STX    2,S      SAVE X REG IN STK AS NEW PUNCH ADD
FCB3 AC   E4            CMPX   ,S       COMPARE TO END ADDR
FCB5 26   C3            BNE    PUNCH2   $FCB5 PUNCH NOT DONE, CONT
FCB7 86   14    PUNEXT  LDA    #$14     LOAD "DC4" PUNCH OFF CODE
FCB9 17   0123          LBSR   OUTCH    OUTPUT IT
FCBC 32   65            LEAS   5,S      READJ STK PTR
FCBE 39                 RTS
```

APPENDIX B

SOFTWARE DRIVERS

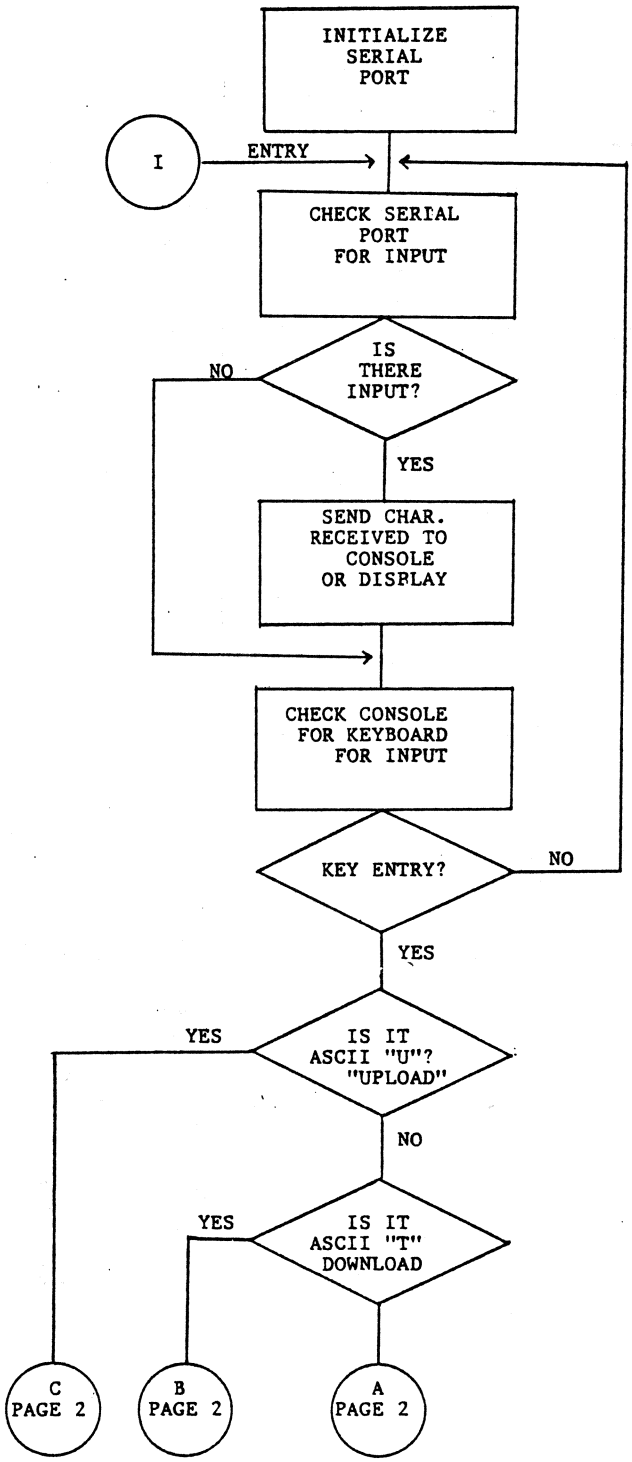
ASCII CODES (HEXIDECIMAL)

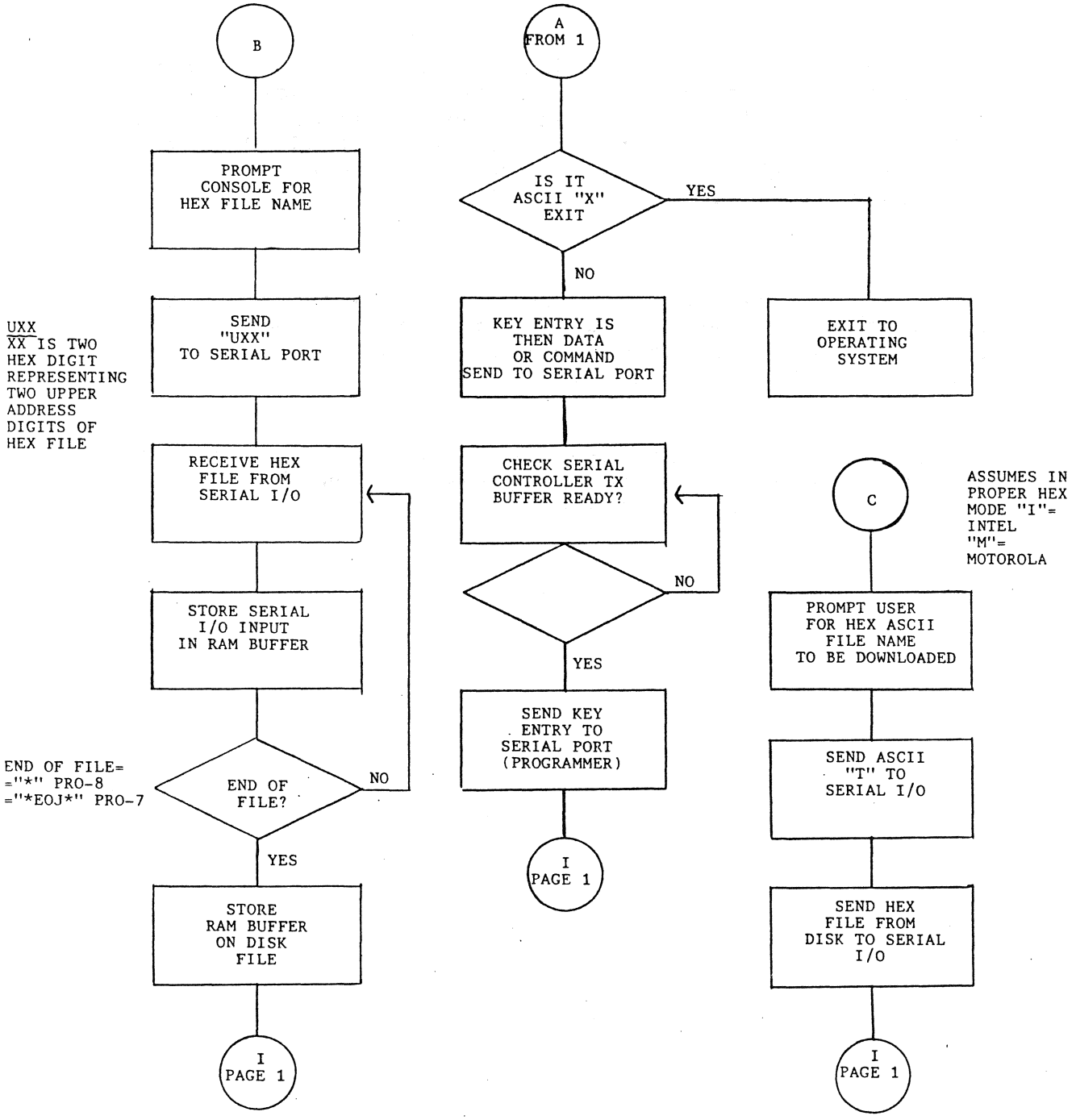
				B7	0	0	0	0	0	0	0	0
				B6	0	0	0	0	1	1	1	1
				B5	0	0	1	1	0	0	1	1
				B4	0	1	0	1	0	1	0	1
BINARY				HEX								
B3	B2	B1	B0	$	0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE	SP	0	@	P	\	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STK	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	\	7	G	W	g	w
1	0	0	0	8	BS	CAN	(	8	H	X	h	x
1	0	0	1	9	HT	EM	)	9	I	Y	i	y
1	0	1	0	A	LF	SUB	*	:	J	Z	j	z
1	0	1	1	B	VT	ESC	+	;	K	[	k	{
1	1	0	0	C	FF	FS	,	<	L	\	l	|
1	1	0	1	D	CR	GS	-	=	M	]	m	}
1	1	1	0	E	SO	RS	.	>	N	^	n	~
1	1	1	1	F	S1	US	/	?	O	_	o	RUBOUT

NUL	NULL
SOH	START OF HEADING
STX	START OF TEXT
ETX	END OF TEXT
EOT	END OF TRANSMISSION
ENQ	ENQUIRY
ACK	ACKNOWLEDGE
BEL	BELL
BS	BACK SPACE
HT	HORIZONTAL TABULATION
LF	LINE FEED
VT	VERTICAL TABULATION
FF	FORM FEED
CR	CARRIAGE RETURN
SO	SHIFT OUT
SI	SHIFT IN

DLE	DATA LINK ESCAPE
DC1	DEVICE CONTROL 1
DC2	DEVICE CONTROL 2
DC3	DEVICE CONTROL 3
DC4	DEVICE CONTROL 4
NAK	NEGATIVE ACKNOWLEDGE
SYN	SYNCHRONOUS IDLE
ETB	END OF TRANSMISSION
CAN	CANCEL
EM	END OF MEMORY
SUB	SUBSTITUTE
ESC	ESCAPE
FS	FILE SEPARATOR
GS	GROUP SEPARATOR
RS	RECORD SEPARATOR
US	UNIT SEPARATOR

GENERALIZED SOFTWARE DRIVER FLOW CHART




B
PROMPT CONSOLE FOR HEX FILE NAME
SEND "UXX" TO SERIAL PORT
UXX
XX IS TWO HEX DIGIT REPRESENTING TWO UPPER ADDRESS DIGITS OF HEX FILE
RECEIVE HEX FILE FROM SERIAL I/O
STORE SERIAL I/O INPUT IN RAM BUFFER
END OF FILE= ="*" PRO-8 ="*EOJ*" PRO-7
END OF FILE?
NO
YES
STORE RAM BUFFER ON DISK FILE
I PAGE 1
A FROM 1
IS IT ASCII "X" EXIT
YES
NO
KEY ENTRY IS THEN DATA OR COMMAND SEND TO SERIAL PORT
EXIT TO OPERATING SYSTEM
CHECK SERIAL CONTROLLER TX BUFFER READY?
NO
YES
SEND KEY ENTRY TO SERIAL PORT (PROGRAMMER)
I PAGE 1
C
ASSUMES IN PROPER HEX MODE "I"= INTEL "M"= MOTOROLA
PROMPT USER FOR HEX ASCII FILE NAME TO BE DOWNLOADED
SEND ASCII "T" TO SERIAL I/O
SEND HEX FILE FROM DISK TO SERIAL I/O
I PAGE 1

TRS-8Ø COLOR COMPUTER

DOWNLOAD PROGRAM

This basic program will download from RAM (source) addresses as specified with selectable destination address. The "Device Command Line" as entered is sent in front of the Motorola format file (add "T" for DOWNLOAD) to allow control of the PROM Programmer. The file ends with "S9".

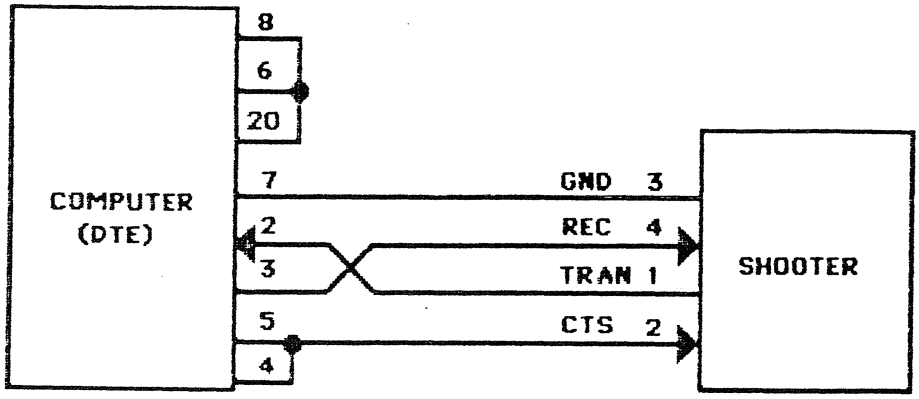
The program as supplied should be compatible with Extended or Non-Extended TRS-8Ø Color Systems. The comment lines show changes which will make the program faster with Extended BASIC.

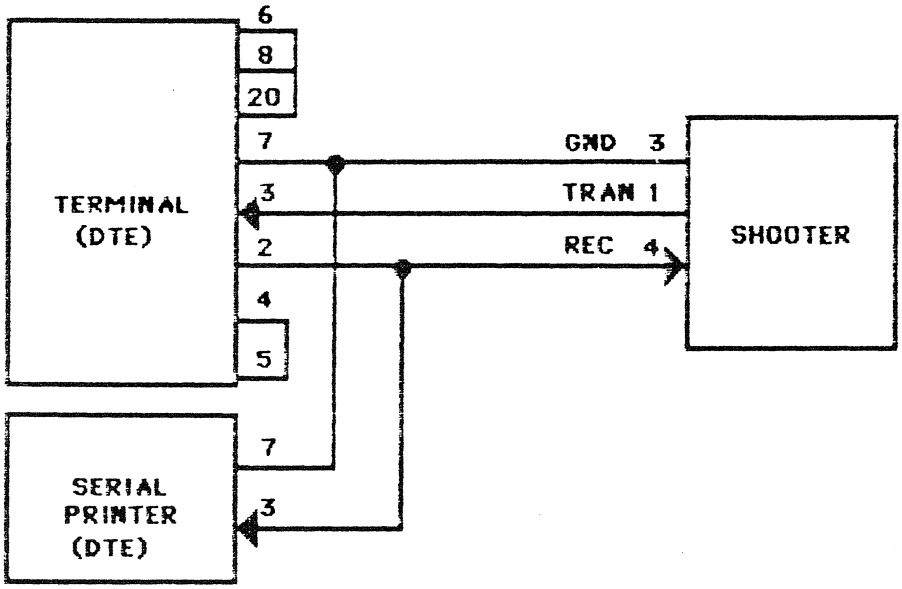
To use your Color Computer as a terminal, you can utilize standard terminal programs such as the "SUPER COLOR" provided by Microware.

```
10 ' PROGRAM TO DOWNLOAD MEMORY FROM COLOR COMPUTER
20 ' MEMORY IN MOTOROLA HEXADECIMAL FORMAT
30 ' THIS PROGRAM IS COMPATIBLE WITH EXTENDED
40 ' OR NON-EXTENDED COLOR BASIC
50 INPUT "BAUD RATE = ";V
60 V=INT(55500/V-4.5)
70 V1=INT(V/256):POKE 149,V1
80 V=V-256*V1:POKE 150,V
85 PRINT #-2, "M"
90 INPUT "SOURCE" STARTING ADDRESS (HEX)= ";V$
100 GOSUB 270:SS=V
110 INPUT "SOURCE ENDING ADDRESS (HEX)= ";V$
120 GOSUB 270:SE=V
130 INPUT "DESTINATION START ADDRS (HEX)= ":V$
140 GOSUB 270:DS=V
150 INPUT "DEVICE COMMAND LINE          =";V$
160 PRINT #-2,V$
165 FOR I = 1 TO 1000: NEXT I
170 L=SE-SS:IF L>15 THEN L=15'DECIDE LINE LENGTH
180 IF L<0 THEN 260
190 V=L+4:CS=0:PRINT #-2,"S1";GOSUB 390'OPEN LINE
200 V=DS:GOSUB 340'DO ADDRESS
210 FOR SS=SS TO SS+L
220 V=PEEK(SS):GOSUB 390:NEXT SS
230 V=256*INT(CS/256)-CS+255
240 GOSUB 390:PRINT #-2
250 DS=DS+L+1:GOTO 170
260 PRINT #-2,"S9030000FC":END
270 ' HEX STRING TO VALUE SUBROUTINE
280 ' FOR EXTENDED COLOR BASIC,USE INSTEAD
290 ' V=VAL("&H"+V$):RETURN
300 V=0:FOR I=1 TO LEN(V$)
310 V1=ASC(MID$(V$,I,1))-48
320 IF V1>16 THEN V1=V1-7
330 V=V*16+V1:NEXT I:RETURN
340 ' PRINT 4 HEX DIGITS SUBROUTINE
350 V1=INT(V/256):V2=V-256*V1:V=V1
360 GOSUB 390 'DO 2 DIGITS
370 V=V2:GOSUB 390 'DO OTHERS
380 RETURN
390 ' PRINT 2 HEX DIGITS SUBROUTINE
400 CS=CS+V 'DO CHECKSUM
410 ' WITH EXTENDED BASIC, NOW USE
420 ' IF V<16 THEN PRINT #-2,"0";
430 ' PRINT #-2,HEX$(V);:RETURN
440 V1=INT(V/16):V=V-16*V1
450  GOSUB460: V1=V: GOSUB460:RETURN
460 PRINT #-2,MID$("0123456789ABCDEF",V1+1,1);:RETURN
```

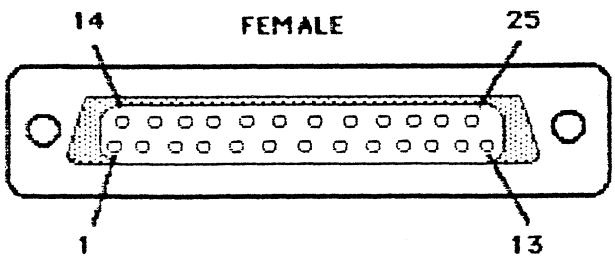
APPENDIX C

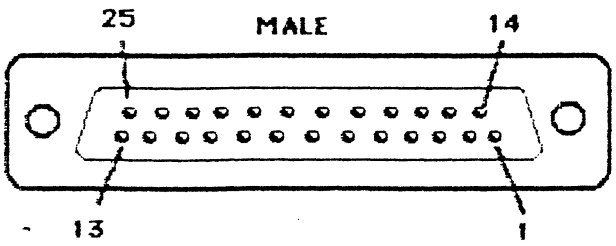
SERIAL CONNECTIONS



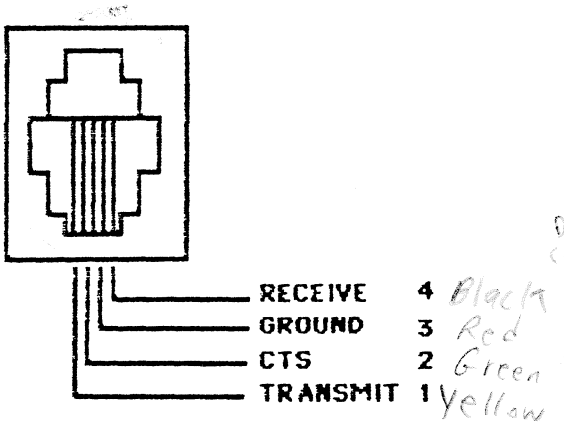


DB-25 CONNECTORS





RJ-11



APPENDIX D

EPROMS

GENERAL EPROM/EPROM PROGRAMMING HANDBOOK

WHAT ARE EPROMs?

This question may be a very obvious to almost anyone who has puchased a programmer. However for the very recent to the field, EPROMs are MOS semiconductor memories used mostly in conjunction with a microprocessor. The instructions and data for the micro (Z-80, 8080, 6800 as examples) are stored permanently in the EPROM. EPROM is the acronym for ERASABLE PROGRAMMABLE READ ONLY MEMORY. Once a UV EPROM has been programmed, it can only be erased by an ULTRAVIOLET source (Logical Devices QUV-T8 or equivalent). For this reason most computers use EPROMs to store vital computer instructions that cannot be lost after power-down. Since EPROMs cannot be written to by the microprocessor directly, the computer system will need RAM and DISK memory for temporary storage as well as application data and programs.

Many companies use EPROMs to store application progams usch as GAMES or computer utility programs on EPROMs packaged in cartridges. These EPROMs contain machine language instructions for the particular microprocessor used in the computer. User often need not to be concerned with the contents of the EPROMs. However, for the technically inclined, who want to decipher the code and make his version of the program, the EPROM must be read into an EPROM PROGRAMMER (such as the Logical Devices PROMPRO series). Once the EPROM has been read into the programmer, the data can be sent-up to a computer from the serial port for storage on the disk drive.

At this point user must be very familiar with the language of the microprocessor. The uploaded file can be disassembled by a software development tool called a "DISASSEMBLER". This process converts the HEX machine instructions to a series of text instruction mnemonics representing the program code. This assembled code can be entered in a text editor or a word processor for alterations. Once the changes have been made to the code. The program can be converted back to machine code level by using a software development tool called an "ASSEMBLER". The output of the assembler that is generally either in an absolute HEX or in a formatted HEX form is stored back on the disk. This file now is to be sent back to the programmer by an operation called "DOWNLOAD". The EPROM programmer generally accepts a file in a formatted HEX form. There are two industry standards; INTEL HEX or MOTOROLA HEX. Both formats are similar. After the PROMPRO programmer receives the ASCII HEX file, it automatically extracts the data section (Machine HEX Code) and stores it in its

RAM buffer. A blank EPROM (one that contains all "1" pattern) is placed on the programmer socket and the RAM data is programmed on the EPROM.
After program operation the EPROM is removed and placed either in the cartridge or the EPROM socket inside of the computer.
Hopefully that you did not hmake any mistakes in writing you program, and after you have powered up your computer everything works to you satisfaction.

You must make sure that anytime that you deal with another company's software program, that you understand all of the copyright laws and by altering those programs you are not violating any of those laws.

EPROMs are fabricated in a variety of memory sizes, however often in an 8-bit organization. The standard memory sizes available today are the following:

GENERIC TYPE	MEMORY SIZE	PACKAGE
2716	2048 X 8	24 PIN
2732	4096 X 8	24 PIN
2764	8192 X 8	28 PIN
27128	16384 X 8	28 PIN
27256	32768 X 8	28 PIN

EPROMs also are available with several choices of access time speeds. The dash number after the generic type designation on the EPROM often indicates the speed. You must consult the data sheet for the particular manufacturer to insure that the EPROM you are using meets the access-time required by your computer. The PROMPRO-8 EPROM programming equipment is indifferent to the EPROM access-time and can program any speed EPROM.

EPROMs can be erased and reprogrammed many times, although as the number of eraser time increases so does the time required to erase them. A new EPROM with a good quality Quartz window can take as little as seven minutes to erase with a QUV-T8/Z EPROM eraser.

Most of the EPROMs today operate with a power supply voltage (VCC) of 4.75 - 5.25 with 5.00 nominal value. VCC is spplied to pin 24 for 24 pin packages and pin 28 for 28 pin packages.

In order to program an EPROM, it is required to supply a high voltage to one of the signal pins (PROG) and pulse one other or the same pin for a period of 50 milliseconds. While applying the proper address and data to the EPROM.

EPROMs are extremely sensitive to static electricity. It is imperative that you always use conductive foam pad when

transporting EPROMs. You must also use a grounding strap or grounding bar whenever working with EPROMs or the programming equipment directly.

Most EPROMs, when shipped from the factory, are already erased (all FF pattern). It is a good idea however, to check all EPROMs for the blank pattern prior to programming.

EPROMs are also very sensitive to power glitches. If either severe storms or unusually high power interruptions are occurring in your area, it is advisable to postpone PROM programming until conditions improve. Remove EPROMs from the equipment during power up and power down operations.

FADING

Fading occurs on EPROMs that are either poorly programmed or defective. EPROMs require a certain minimum number of electrical charges in the floating gate of the FET switches. These charges tend to gradually leak out over an extended period of time, if the EPROM programming equipment does not supply the required amount of charges. The EPROM may initially program and verify and even work in the circuit for several months until the EPROM bits start intermitantly changing state. Some EPROM programming equipment manufacturer's may promise fast programming time on EPROMs to sell their equipment without consideration to the proper programming algorithms. Logical Devices, Inc. very strongly adheres to the manufacturer's specifications to maximize both the field reliability and the longevity of your EPROMs.

EPROM TYPES

Within the same generic family there may be several types of EPROMs that very as far as the programming voltage requirements. Presently there are two common programming voltage levels - 25 volts and 21 volts. Some newer devices require a 12.5 programming voltage.

An example of this is the 2732 EPROM that requires a 25 volt programming voltage while a 2732A EPROM requires a programming voltage of 21 volts. Trying to program using the wrong programming voltage can destroy the device.

FAILURES

EPROMS have a much better programming yield than Bipolar PROMs. A user must have a 5% or less programming yield failure rate for EPROMS. If the failure rate exceeds 5%, the user should suspect either the EPROMs or the equipment.

PURCHASING EPROMs

Logical Devices' recommends that you select reliable sources for your components. To avoid problems, refrain from buying EPROMs from either uncertain origins or questionable surplus establishments. Leading reputable sources are:

Hamilton Avnet
Hallmark
Arrow Electronics
Hammond Electronics

Consult your local telephone directory for local branch offices of the above distributors.

EPROM PINOUTS

MANUFACTURERS OF PROGRAMMABLE READ-ONLY MEMORIES

ADVANCED MICRO DEVICES
901 THOMPSON PL
SUNNYVALE, CA 94086
(408) 732-2400

AMERICAN MICROSYSTEMS INC
3800 HOMESTEAD RD
SANTA CLARA, CA 95051
(408) 246-0330

ELECTRONIC ARRAYS
550 E MIDDLEFIELD RD
MT VIEW, CA 94043
(415) 964-4321

FAIRCHILD SEMICONDUCTOR
464 ELLIS ST
MT VIEW, CA 94042
(415) 962-5011

FUJITSU MICROELECTRONICS
2945 OAKMEAD VILLAGE CT
SANTA CLARA, CA 95051
(408) 729-1700

GEN INSTRUMENT CORP
MICROELECTRONIC DIV
600 W JOHNS ST
HICKSVILLE, NY 11802
(516) 733-3000

HARRIS SEMICONDUCTOR
BOX 883
MELBOURNE, FL 32901
(305) 724-7407

HITACHI AMERICA LTD
1800 BERING DR
SAN JOSE, CA 95112
(408) 292-6404

HUGHES AIRCRAFT CO
SOLID ST PROD DIV
500 SUPERIOR AVE
NEWPORT BCH, CA 9266
(714) 759-2411

INTEL CORP
3065 BOWERS AVE
SANTA CLARA, CA 95051

INTERSIL INC
10710 N TANTAU AVE
CUPERTINO, CA 95014

MOSTEK CORP
1215 W CROSBY DR
CARROLLTON, TX 75006

MITSUBISHI/MEICO
3030 E VICTORIA ST
COMPTON, CA 90221
(213) 537-7131

MONOLITHIC MEMORIES INC
1165 E ARQUES AVE
SUNNYVALE, CA 94086
(408) 739-3535

MOTOROLA INC
INTERGRATED CIRC DIV
3501 ED BLUESTEIN BL
AUSTIN, TX 78721
(512) 928-6000

NATIONAL SEMICONDUCTOR
2900 SEMICONDUCTOR DR
SANTA CLARA, CA 95051
(408) 737-5000

NEC MICROCOMPUTERS INC
173 WORCESTER ST
WELLESLEY, MA 02181
(617) 239-1910

NITRON INC
10420 BUBB RD
CUPERTINO, CA 95014
(408) 255-7550

OKI SEMICONDUCTOR
SUITE 405
SANTA CLARA, CA 95051
(408) 984-4842

PLESSEY SEMICONDUCTOR
1641 KAISER AVE
IRVINE, CA 92714
(714) 540-9979

PANASONIC
1 PANASONIC WAY
SECAUCUS, NJ 07094
(201) 348-7276

RAYTHEON SEMICONDUCTOR
350 ELLIS ST
MT VIEW, CA 94042
(415) 968-9211

SGS-ATES SEMICONDUCTOR CRP
240 BEAR HILL RD
WALTHAM, MA 02154
(617) 890-6688

RCA SOLID STATE DIV
RTE 202
SOMERVILLE, NJ 08876
(201) 685-6000

SIGNETICS CORP
811 E ARQUES AVE
SUNNYVALE, CA 94086
(408) 739-7000

TEXAS INSTRUMENTS INC
BOX 225012, M/S 308
DALLAS, TX 75265

SYNERTEK
BOX 552
SANTA CLARA, CA 95051
(408) 988-5611

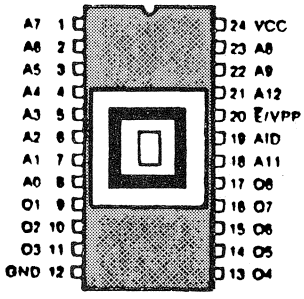
TOSHIBA AMERICA INC
2151 MICHELSON DR
IRVINE, CA 92715
(714) 955-1155

XICOR INC
1221 INNSBRUCK DR
SUNNYVALE, CA 94086
(408) 734-3041

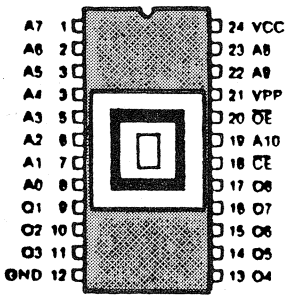
ZILOG
10460 BUBB RD
SUNNYVALE, CA 94086
(408) 446-4666

EPROM PIN-OUTS

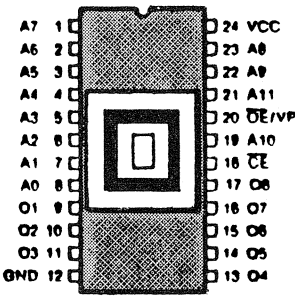
MCM 68764, 68766



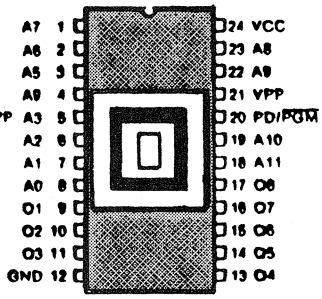
2516, 2716, 27C16



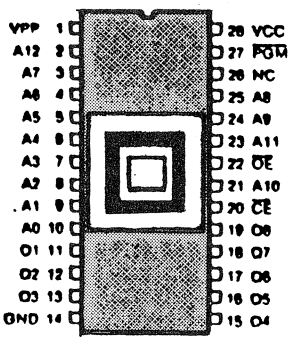
2732, 27C32, 2732A



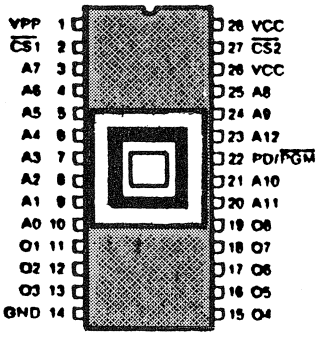
25L32, 2532



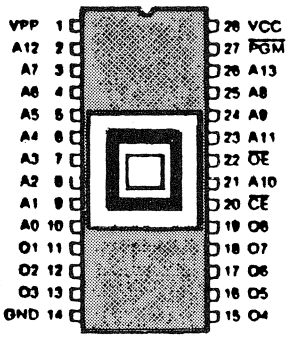
2764, 27C64, 2764A



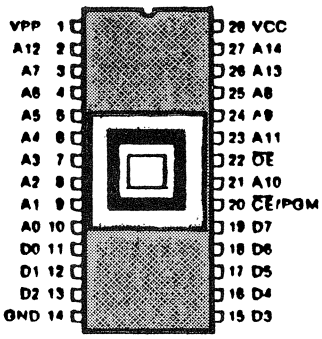
2564



27128, 27128A



27256



EPROM PIN-OUT GUIDE

EPROM TYPE	VCC PIN	GRND PIN	VPP PIN	VPP VOLTAGE	PGM PULSE	PGM WITH PULSE	PIN OUTPUT ENABLE
2716	24/26	12/14	21/23	25V ± 1V	18/20	50ms	20/22
2732	24/26	12/14	20/23	25V ± 1V	20/22	50ms	20/22
2732A	24/26	12/14	20/22	21V ± 1V	20/22	50ms	20/22
2764	28	14	1	21V ± v	27	50ms	20/22
2564	28	14	1	21V ± V	22	50ms	20/22
27128	28	14	1	21V ± V	27	50ms	20/22
27256	28	14	1	12.5V ± 5V	27	*	20/22
68764	24/26	12/14	20/22	25V ± 1V	20/22	*	20/22
2532	24/26	12/14	20/22	25 ± 1V	20/25	50ms	20/22

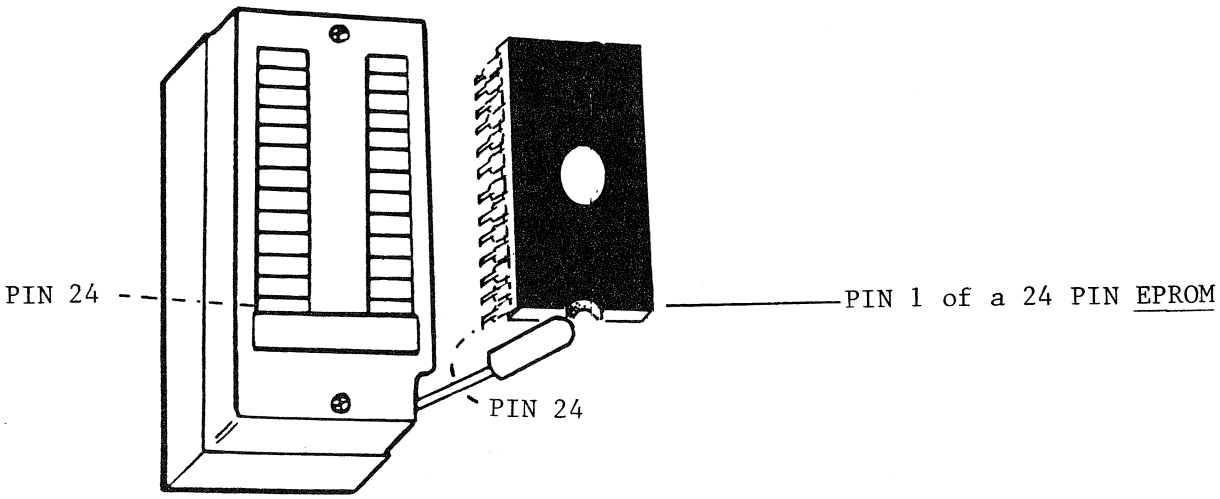
PIN NUMBERS ARE BOTH INDICATED FOR BOTH 24 PIN AND 28 PIN SOCKETS BY THE "/".

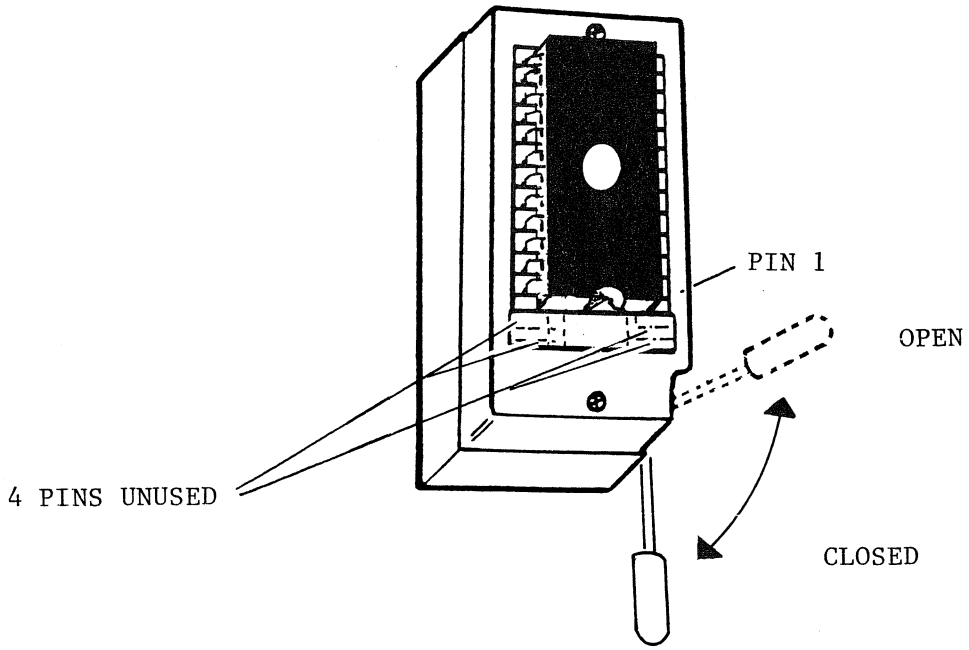
2764A, 27128A 12.5 VPP
VPP = PROGRAM VOLTAGE
VCC = SUPPLY CURRENT 5V ± .25V

APPENDIX E

HOW TO INSERT EPROMS IN THE ZIF SOCKETS

INSERTING A 24 PIN DEVICE IN SOCKET A





ZIF SOCKET 1

1.0 DESCRIPTION

To avoid the mis-socketing of 24-pin EPROMs in the 28-pin ZIF socket, a ZIF-LOCK is provide with this EPROM Programmer. When properly installed, the ZIF-LOCK effectively covers pins 1, 2, 27, and 28 of the 28-pin ZIF socket.

1.1 INSTALLATION

NOTE: Refer to Figure 1 when installing the ZIF-LOCK.

1. Be sure the ZIF Socket handle is in the up (release) position.

2. Snap the ZIF-LOCK in place over pins 1, 2, 27, and 28 by first inserting one end and then the other. The ZIF-LOCK should snap firmly into place.

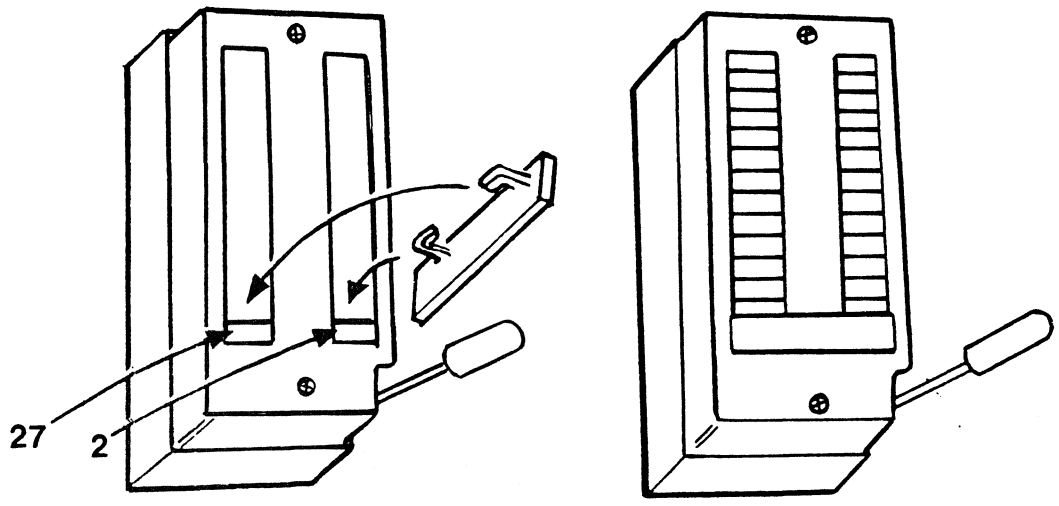


Figure 1. ZIF-LOCK Installation

1.2 REMOVAL

----- CAUTION -----

Be sure the ZIF socket handle is in the release position.

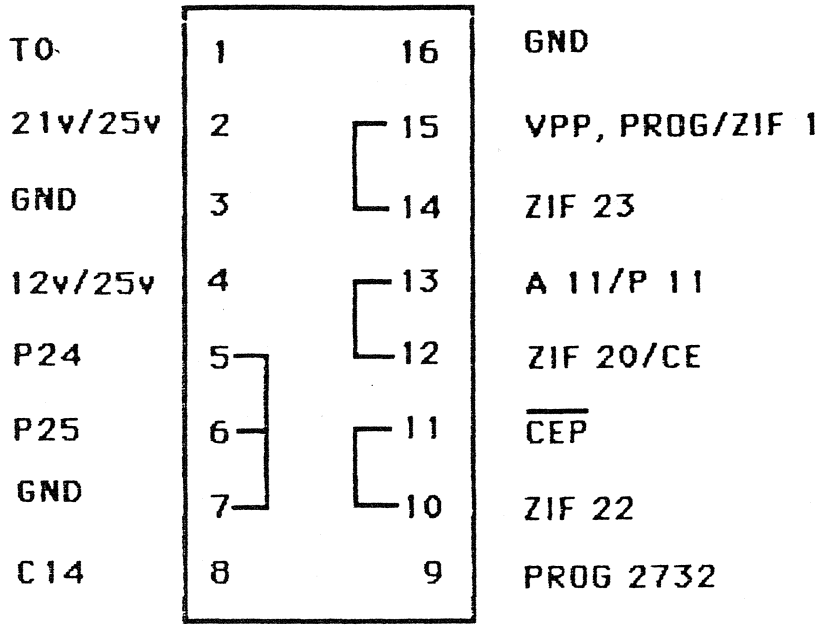
1. Remove the ZIF-LOCK simply by lifting one end and then the other. The ZIF-LOCK should lift easily from the socket.

APPENDIX F

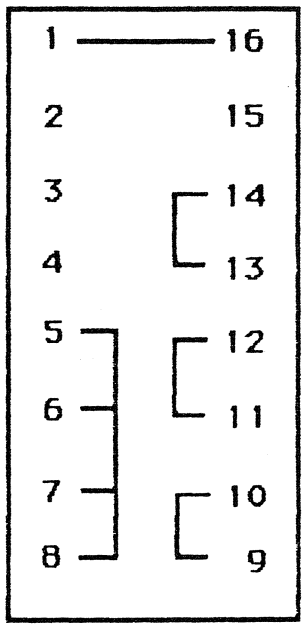
CONFIGURATORS

STANDARD CONFIGURATOR SET

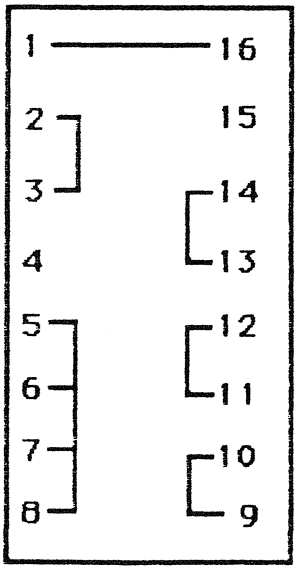
2716



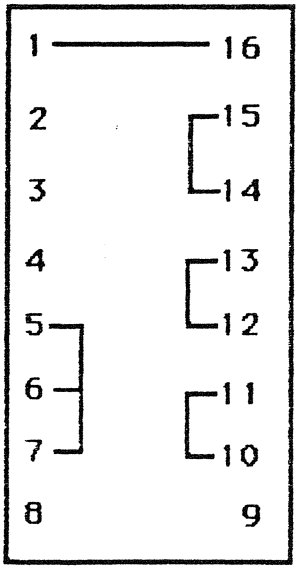
2732



2732A



2532



STANDARD CONFIGURATOR SET