64000 LOGIC DEVELOPMENT SYSTEM

PROM PROGRAMMER REFERENCE MANUAL

HEWLETT PACKARD

CERTIFICATION

Hewlett-Packard Company certifies that this product met its published specifications at the time of shipment from the factory. Hewlett-Packard further certifies that its calibration measurements are traceable to the United States National Bureau of Standards, to the extent allowed by the Bureau's calibration facility, and to the calibration facilities of other International Standards Organization members.

WARRANTY

This Hewlett-Packard system product is warranted against defects in materials and workmanship for a period of 90 days from date of installation. During the warranty period, HP will, at its options, either repair or replace products which prove to be defective.

Warranty service of this product will be performed at Buyer's facility at no charge within HP service travel areas. Outside HP service travel areas, warranty service will be performed at Buyer's facility only upon HP's prior agreement and Buyer shall pay HP's round trip travel expenses. In all other cases, products must be returned to a service facility designated by HP.

For products returned to HP for warranty service. Buyer shall prepay shipping charges to HP and HP shall pay shipping charges to return the product to Buyer. However, Buyer shall pay all shipping charges, duties, and taxes for products returned to HP from another country.

HP warrants that its software and firmware designated by HP for use with an instrument will execute its programming instructions when properly installed on that instrument. HP does not warrant that the operation of the instrument, or software, or firmware will be uninterrupted or error free.

LIMITATION OF WARRANTY

The foregoing warranty shall not apply to defects resulting from improper or inadequate maintenance by Buyer, Buyer-supplied software or interfacing, unauthorized modification or misuse, operation outside of the environmental specifications for the product, or improper site preparation or maintenance.

NO OTHER WARRANTY IS EXPRESSED OR IMPLIED. HP SPECIFICALLY DISCLAIMS THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

EXCLUSIVE REMEDIES

THE REMEDIES PROVIDED HEREIN ARE BUYER'S SOLE AND EXCLUSIVE REMEDIES. HP SHALL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WHETHER BASED ON CONTRACT, TORT, OR ANY OTHER LEGAL THEORY.

ASSISTANCE

Product maintenance agreements and other customer assistance agreements are available for Hewlett-Packard products.

For any assistance, contact your nearest Hewlett-Packard Sales and Service Office.

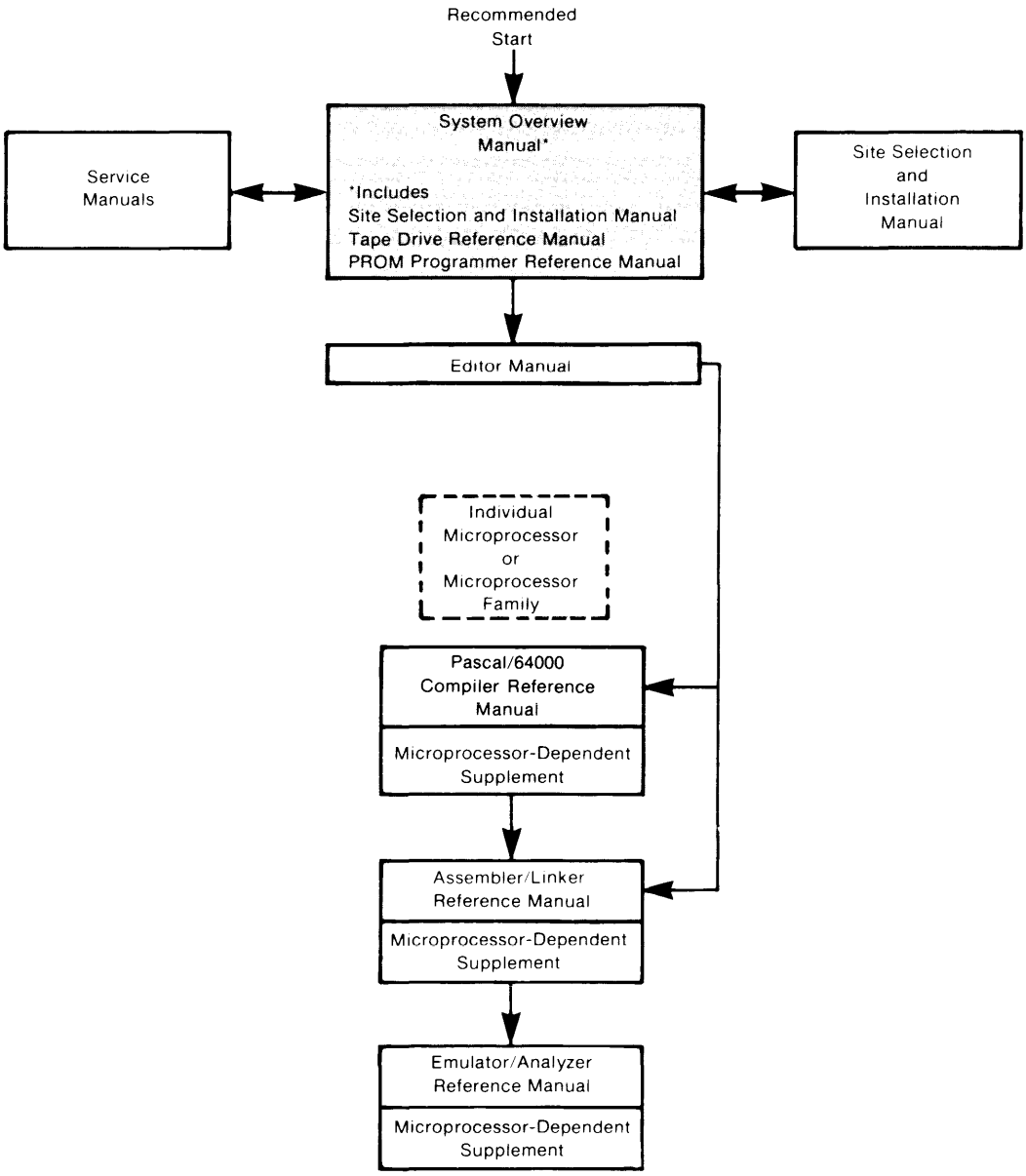
CW&A 9/79

Model 64000 Reference Manuals

The following block diagram shows the documentation scheme for the HP Model 64000 Logic Development System. The interconnecting arrows show the recommended progression through the manuals as a way of gaining familiarity with the system.

For a detailed list showing specific manuals and their part numbers, refer to the System Overview Manual.

Manual Map



Printing History

Each new edition of this manual incorporates all material updated since the previous edition. Manual change sheets are issued between editions, allowing you to correct or insert information in the current edition.

The part number on the back cover changes only when each new edition is published. Minor corrections or additions may be made as the manual is reprinted between editions.

First Printing..... September 1980 (Part Number 64500-90901)
Second Edition.....February 1981 (Part Number 64500-90904)
Third Edition..... July 1981 (Part Number 64500-90906)

PROM Programmer
Reference Manual

HP Model 64000 Logic Development System

NOTE

The contents of this manual have been taken directly from the System Overview Manual and reflect the chapter number, page numbers, and figure and table numbers used in that manual.

Chapter 11
PROM Programmer

Introduction

This chapter provides instructions for installing and operating the Model 64000 PROM Programmer option. The first part of this chapter describes the Model 64000 PROM Programmer option and shows you how to install it in a Model 64000 terminal. There are two versions of the Model 64000 PROM Programmer: the one used in disc-based systems, and the one used in tape-cartridge based systems. This section describes the operating procedures for each of the PROM programmer versions. All of the error messages that can be displayed by either PROM programmer version are listed and described at the end of this chapter.

Description

The Model 64000 PROM Programmer option provides the capability to program, verify, and read most popular bipolar and MOS (fusible link, UV, or electrically erasable) PROM's. Each option includes a 64502A PROM Interface Module, a 64501A Control Card, a special cable (HP Part No. 64501-61601), and a choice of unique personality PROM modules. The PROM Programmer option always includes circuitry for the operator self test and for the service self test.

Table 11-1 lists power supply requirements of the PROM Programmer option. This information will help you determine how many circuit cards can be loaded into the card cage of your terminal.

Table 11-1. PROM Programmer Power Requirements

Power Supply	+40V	+12V	+5V	-5.2V	-12V
Typical Current	0.38A	.03A	1.07A	.01A	.01A

CAUTION

Before applying or removing system power, make certain there are no PROMs or other devices installed in any of the module sockets. To do otherwise may cause damage to the device or may alter its program.

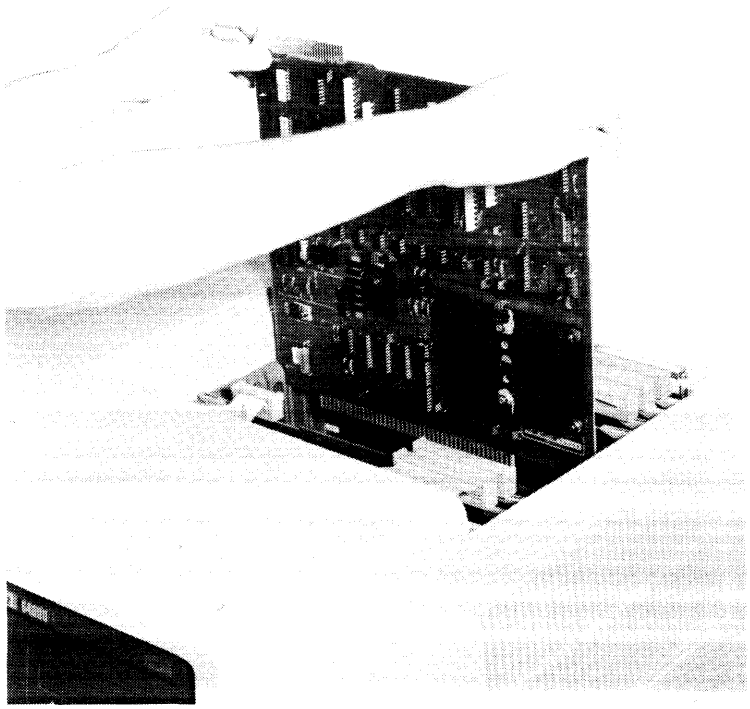


Figure 11-1. Control Board Installation

Installation

CAUTION

Turn off station power before installing PC card or connecting personality module.

Control Board Installation

The Model 64501A driver card must be installed in the card cage. Remove top lid by loosening the two screws in the rear of the top cover. Lift the back portion of the lid, pull slightly to the rear and remove. Install the card (component side toward front of instrument) in any card slot 0 thru 9 (see figure 11-1).

NOTE

To minimize cable lengths and problems, it is recommended that the PC card be placed in slot 1.

Connect the single-keyed connector of the PROM cable (HP Part No. 64501-61601) to the top left-hand connector of the circuit board.

Run the cable across the top of the card cage and down behind the right side of the front panel. Both connectors on the end of the cable should be visible in the well on the right side of the Model 64100A.

Personality Module Installation

Personality modules come with one or two connectors (see figure 11-2). The cable is designed for use with either type. If the module has only one connector, it is best to connect the end cable connector to the module so the module will seat fully into the well. If the personality module has two connectors, connect both cable connectors.

NOTE

The cable and module connectors are keyed so there is only one way to connect them.

After the cable is connected, place the module in the well and adjust it until it is flush. Turn the hold-down screw clockwise until it is tight.

PROM Installation

Pin 1 is identified on each personality module. Place the desired PROM in the socket so that pin 1 is aligned with pin 1 of the socket. Press the socket lever down to make good contact with all PROM pins.

CAUTION

Certain PROMs can be damaged by static electricity. Observe the manufacturer's handling procedures to prevent damage. Verify that the correct PROM programmer module is installed in the Model 64000.

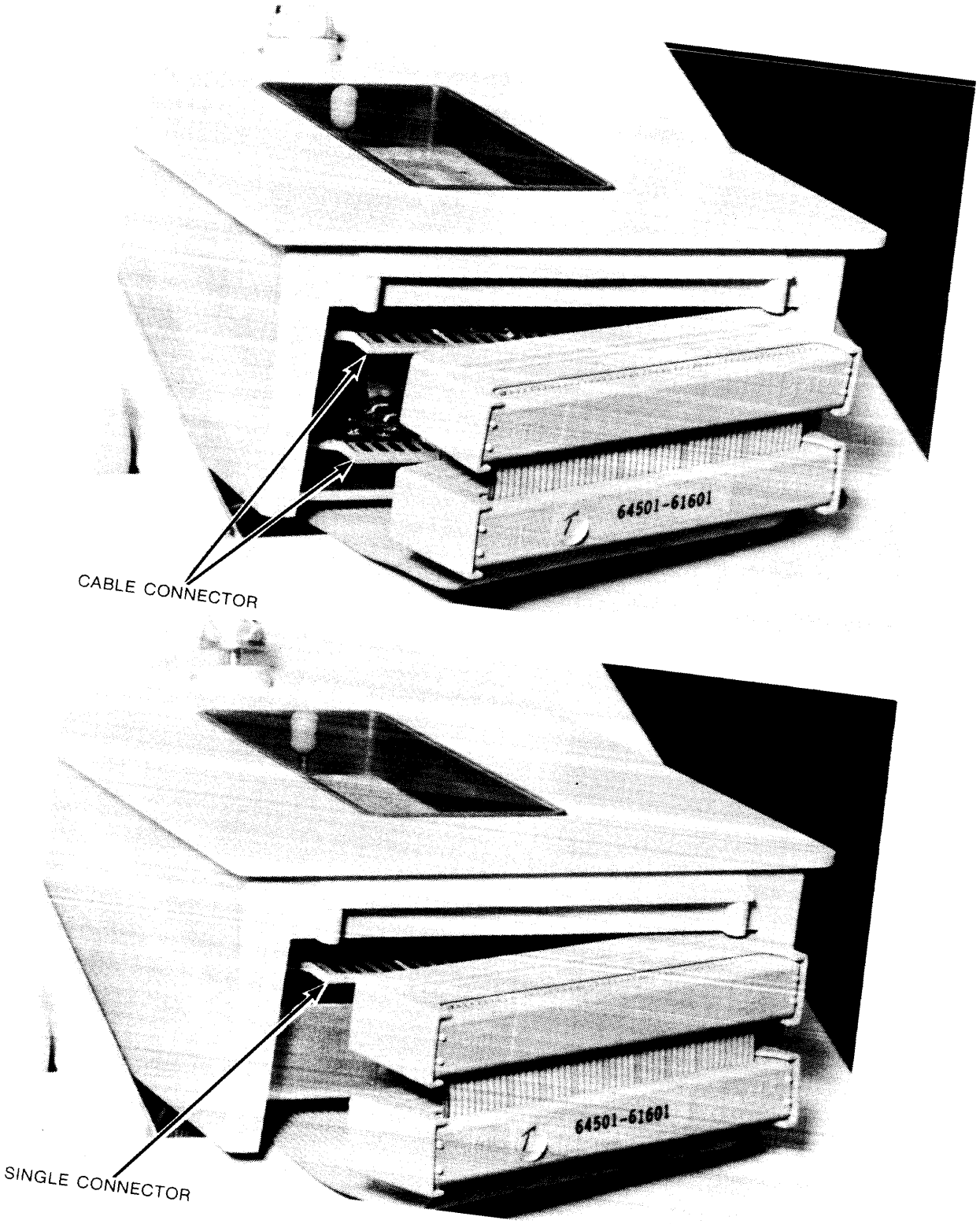


Figure 11-2. Personality Module Installation

Operating the Disc-Based PROM Programmer

The following information describes how to operate the disc-based PROM Programmer. This information begins by listing all of the parameters that are common to the softkeys that appear during disc-based PROM Programmer operation. Then each of the softkeys is described in detail.

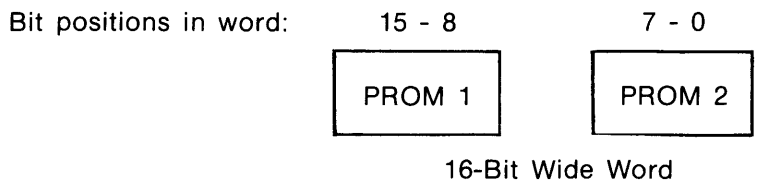
Parameters

The following parameters are used by many of the softkeys that appear during disc-based PROM Programmer operation.

<**ADDRESS**> The <ADDRESS> parameter specifies where the terminal should begin its display of PROM data. If you make no entry in this field, the terminal will begin its display with the data at address 0H. If you make an entry in this field, the top line of the display will include the data at the address you selected.

START <**value**> The start parameter specifies a starting address in the absolute or listing file where programming or reading should start. The default value is 0.

BIT <**value**> The bit parameter allows bytes and nibbles to be extracted from a file whose word size is greater than the PROM size. The bit number given is the least significant bit of the word in the file data. For instance, if a processor word is 16 bits wide and memory is going to be two 8-bit PROMs, the bit number allows programming of these PROMs from the same file as follows:



program_from FILE bit 8 PROM 1, bits 15 - 8

program_from FILE bit 0 PROM 2, bits 7 - 0

The default value of bit number is 0 (the right-most bit in the word).

ROM_ADDR The PROM address function permits programming or reading a sub range of the PROM. Normally, programming or reading will affect the whole address range of the PROM, but there are occasions when we only want to affect a sub range in the PROM. The rom_addr can specify a starting point and an ending address range in the PROM for programming or reading. In the case of rom_addr 5 thru 8, for example, programming or reading will occur in PROM addresses 5 thru 8 inclusive.

NEG_DATA If the system hardware is constructed using negative logic, it will expect the data to be complemented. In such cases, the neg_data option will perform a ones complement of the data before it is programmed into the PROM and will complement data read from the PROM.

NEG_ADDR Again, if the hardware expects an address to be negative logic, the programmer can store data at the ones complement of the address given. For example, if data would normally be programmed at PROM address 0, the programmer would store it at the ones complement of the address (in this case, the maximum PROM address).

No_Blank_Check The PROM Programmer will perform a blank check of the PROM before programming begins. Therefore, if your PROM is not blank but you still want to program it, you can override the blank check feature by using the no_blank_check instruction.

Files used by Disc-Based PROM Programmer

The disc-based PROM Programmer can use two types of files for programming or verifying PROM contents: absolute files and listing files. These two file types are described below.

Absolute files are created by the system linker. They contain information about the processor, the addresses where data is located in memory, and who generated the file. All of this information will be transparent to the user. The programmer is compatible with any microprocessor, regardless of word size.

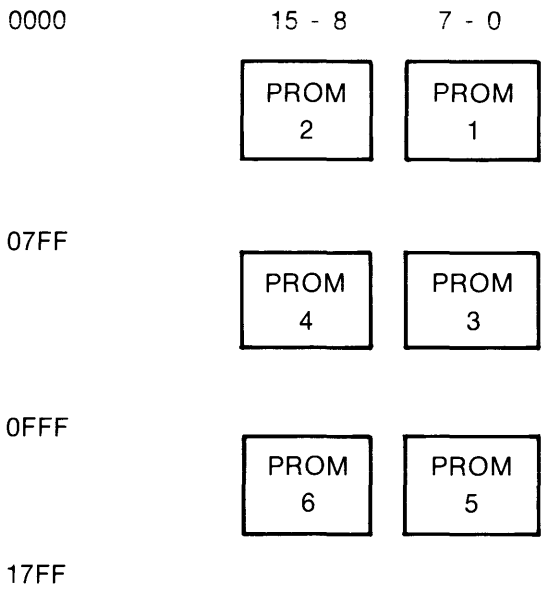
Listing files provide several features within the programmer. For instance, they allow the user to see what is in a PROM. By listing a PROM program to a file, it can be reviewed for changes or corrections via the system editor and returned to the programmer for downloading onto a PROM.

CAUTION

Syntax of the listing file is very important to the PROM Programmer. Be careful, therefore, not to change the basic structure of the file.

The list function has another important feature. It allows the user to generate a memory map which may consist of more than one ROM and cover a 32-bit address space. A list file containing bit widths up to 128 bits wide can be constructed using the bit number function. Also, by specifying the starting address, PROM data can be placed anywhere in the list file, and the existing list data will automatically be updated. For example, if we had a memory which was 6K long by 16 bits wide and we wanted to construct a file reflecting the memory shown below, we would issue the following commands to construct the list file:

```
list_rom to LIST start 0000H bit 0        for PROM #1
list_rom to LIST start 0000H bit 8        for PROM #2
list_rom to LIST start 0800H bit 0        for PROM #3
list_rom to LIST start 0800H bit 8        for PROM #4
list_rom to LIST start 1000H bit 0        for PROM #5
list_rom to LIST start 1000H bit 8        for PROM #6
```



Relationship of the File Address and PROM Address

There are three basic formats for programming a PROM with the PROM Programmer. The three formats are shown in figures 11-3 through 11-5 and discussed in the following paragraphs. All of the programming functions discussed are transparent to the user, but some explanation of how the PROM Programmer handles each of these formats will be helpful.

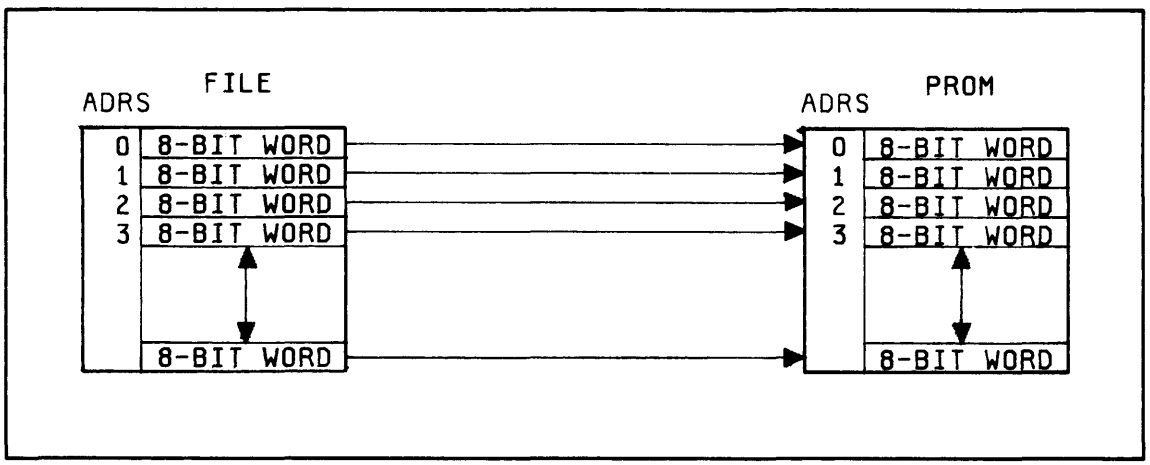


Figure 11-3. Programming a Single PROM where the File Addresses are the same as the PROM Addresses

Figure 11-3 shows the programming of a PROM where the word size in the file is the same as the word size of the PROM. In this situation, you insert the PROM in the Programmer Module, press the **program** softkey, and type in the name of your file. When you press RETURN, the terminal will automatically take the data word from file address 0 (bit-by-bit) and program it into PROM address 0. Then it will take the word from file address 1 and program it into PROM address 1. It will continue this process until it reaches the end of the file.

Parameters are available in the PROM Programmer to help you overcome special programming problems. The parameters can be used as follows:

a. In the case where your file is too large to be contained in one PROM, you will have to distribute blocks of your file over several PROM's. You can program your first PROM from file address 0, but subsequent PROM's will have to be programmed from addresses located further into the file. The **start** <value> parameter is provided to help you accomplish this task. When you include a **start** <value> in your command, the first word taken from the file will not come from address 0. Instead, it will come from the file address you specify in <value>. The next word will be taken from <value>+1, the word after that from <value>+2, etc.

b. There is also a parameter that lets you program a portion of the available space in the PROM without programming the entire PROM. You can specify the address where programming will start in the PROM. You can also format the command as <value> thru <value> if you want to program a limited portion of the center of the PROM without affecting the content on either end of the PROM. The parameter to use for this requirement is the {rom_addr} parameter. If you include a {rom_addr} <value> in your command, the first word taken from your file will be programmed into the PROM at PROM address <value>. The next word taken from your file will be programmed into PROM address <value>+1, etc.

c. You can use the {start} and {rom_addr} parameters together to program activity starting at any file address and any PROM address.

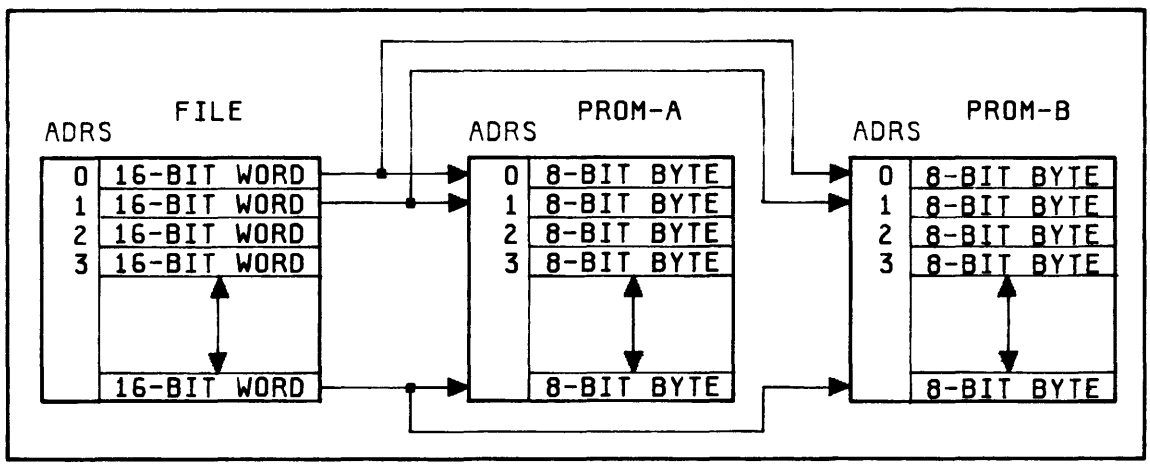


Figure 11-4. Programming Multiple PROM's where the File Address is the same as the PROM Address

When you are programming a set of PROM's like those shown in figure 11-4, you can only program one of the PROM's at a time (you might program PROM-A first and then program PROM-B next). To accomplish this programming, you will need to use the {bit} <value> parameter. To program PROM-A, you can simply specify a command of {program} <the name of your file>, [RETURN]. The PROM Programmer will program bits 0 through 7 of each word into the PROM and send bits 8 through 15 of each word to a location unrecognized by the Programmer Module.

Then you will need to add the bit <value> parameter to your command before you can program the next PROM. This time your command line will read program <the name of your file> bit <8> RETURN . This command will cause the PROM Programmer to send the first eight bits of each word to a location unrecognized by the Programmer Module and begin programming into the PROM with bit 8 of each word. The PROM Programmer reads every bit each time it goes through your file, but it only enters the specified bits into the PROM.

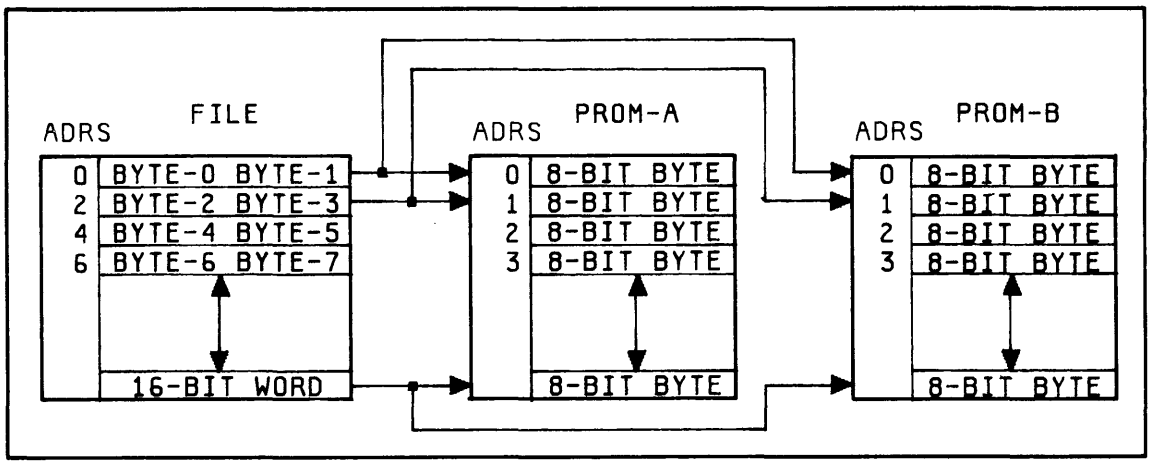


Figure 11-5. Programming Multiple PROM's where the File Address is different from the PROM Address

When you are programming PROM's in a situation like figure 11-5, you may find that the data is programmed into PROM addresses which are different from the addresses you expected to see. As shown above, you would find the first byte of the word at file address 200H programmed into PROM address 100H.

In figure 11-5, the PROM Programmer divides each word in your file into bytes and programs the bytes into consecutive PROM locations. The PROM Programmer accomplishes this task by making multiple passes through the file. On the first pass, the PROM Programmer will take all of the even numbered bytes and program them into PROM-A. It will send the odd-numbered bytes of each word to addresses which are not recognized by the Programmer Module. You must insert the other PROM into the Programmer Module before you start the next pass. Then the PROM Programmer will program PROM-B.

When it finishes programming a PROM address, the PROM Programmer will calculate the next PROM address to be programmed by using the formula shown in figure 11-6. The formula that was chosen to calculate the next address is universal. It will calculate correct PROM addresses for any programming arrangement, regardless of the size of the file word or the size of the PROM bytes. At the end of each pass through the file, the rom_addr parameter is incremented by 1 to begin the next pass through the file at the proper byte address in the file.

$$\text{PROM address} = \left(\text{file address} - \text{file start}\right) \times \left(\frac{\text{address base}}{\text{word size}}\right) + \text{rom\_addr}$$

Where:

PROM address =	next address to be programmed in PROM.
file address =	address in file where next data unit will be obtained.
file start =	first file address where data was taken for PROM programming during this run.
address base =	Smallest unit of data the processor can address.
word size =	Number of bits addressing a data word for the processor.
rom_addr =	PROM address offset where first byte was programmed during the present run.

Figure 11-6. Formula PROM Programmer uses to calculate Next Addresses during programming

To format your commands when programming PROM's that correspond to figure 11-5, use the following procedure:

1. Insert the first PROM to be programmed into the Programmer Module.
2. Enter [program] <the name of your file> [bit] 0, and press [RETURN].
3. PROM-A will be programmed automatically with the correct set of bytes from your file.
4. Remove PROM-A.
5. Insert PROM-B.
6. Enter [program] <the name of your file> [bit] 8, and press the [RETURN] key. PROM-B will be programmed with the corresponding set of bytes from your file.

NOTE

You can program a set of PROM's with more than two PROM's in the set by using the PROM Programmer. Simply repeat steps 2 through 6 of this procedure, and select the correct bit numbers until all of your PROM's are programmed.

PROM Programmer Self-Check

To evoke the PROM programmer self-check, proceed as follows:

Type option_test on command line.

Press 

.

The display will show:

POSITIVE PROM PROGRAMMER TEST
STATUS: Ready for PROM tests

START CYCLE END TEST ____ ____ ____ DAC=10V S/A

The system will display "Digital Test Passed" when the test is completed. This is the extent of the operator's self-check. For a detailed test, refer to the PROM Programmer Service Manual.

Softkeys used by the Disc-Based PROM Programmer

Table 11-2 lists each of the softkeys that is used in the operation of the disc-based PROM Programmer, and describes their functions. On the pages following table 11-2, each of these softkeys is discussed in detail.

Table 11-2. Disc-Based PROM Programmer Softkeys

Softkey Name	Function Selected
program	Causes the terminal to program and verify your PROM from the absolute file or listing file you specify.
verify	Causes the terminal to compare the data in your PROM with the data in the absolute file or listing file you specify.
list	Causes the terminal to create a list of the content of your PROM either on the printer or in a listing file.
check_sum	Causes the terminal to display a 32-bit total of all data stored either in your PROM or in the file you specify.
prog_loc	Causes the terminal to program only the PROM address you specify.
read-loc	Causes the terminal to read only the PROM address you specify.
display	Causes the CRT to display the data at any PROM location you specify.

SYNTAX

program_from <FILE> [parameters]

Parameters

start <value>	starting file address, default 0
bit <value>	lsb of data unit, default entire PROM
rom_addr <value> thru <value>	PROM sub range, default entire PROM
neg_data	negative logic for data, default positive if not given
neg_addr	negative logic for address, default positive if not given
no_blank_check	turns off the automatic blank check feature

FUNCTION

This function programs a PROM from either an absolute or a listing file. Each time a location is programmed, the location is read and compared to the file data. If the comparison does not match, the location is reprogrammed and compared again in an attempt to match the file data. This will occur five times. After the fifth try, the programmer will stop and display a message on the status line stating that the programming function has failed, and programming will stop. If no programming errors occur, the PROM will be programmed and an automatic verification of the PROM will be performed to make sure that programming one location did not affect any others.

SYNTAX

verify_from <FILE> **[**parameters**]**

Parameters

start <value>	starting file address, default 0
bit <value>	lsb of data unit, default 0
rom_addr <value> thru <value>	PROM sub range, default entire PROM
neg_data	negative logic for data, default positive if not given
neg_addr	negative logic for address, default positive if not given
listfile <FILE>	list of locations which fail verify

FUNCTION

The verify command will compare file data with data in the PROM and report the number of failures on the status line. If the PROM verifies, the status will read:

Verify complete, no errors

If the PROM does not verify, the number of failures will be displayed on the status line. Also, if a list file is specified, then a summary of the failures will be listed with PROM address, PROM data, and file data. If the verify does not find any data in the range specified, a warning message will be displayed stating that no locations were verified.

SYNTAX

list<FILE> [parameters]

Parameters

start <value>	starting file address, default 0
bit <value>	lsb of data unit, default 0
rom_addr <value> thru <value>	PROM sub range, default entire PROM
neg_data	negative logic for data, default positive if not given
neg_addr	negative logic for address, default positive if not given

FUNCTION

The list instruction allows the user to obtain a list of PROM contents by listing them to a file or to the printer directly. The list file can then be edited and used to construct memory maps of several PROMs. (Refer to paragraph entitled "File Data".)

SYNTAX

check_sum rom [parameters]

Parameters

rom_addr <value> thru <value>	PROM sub range, default entire PROM
neg_data	negative logic for data, default positive if not given
neg_addr	negative logic for address, default positive if not given

check_sum <file> [parameters]

Parameters

start <value>	starting file address, default 0
bit <value>	lsb of data unit, default 0
rom_addr <value> thru <value>	rom sub range, default entire rom
neg_data	negative logic for data, default positive if not given
neg_addr	negative logic for address, default positive if not given

FUNCTION

The check sum is a 32-bit sum of all data in the specified range in the PROM or from the file.

SYNTAX

program_location <address> to <value>

FUNCTION

This function permits the programming of one location at a time and is useful when dealing with check sums. The address specified must be in the PROM address range, and the lower 4 or 8 bits of the value must be used. The programmer will try to program the location without checking to see if it is blank or not; then it will read the location back and compare data. The status line will tell if the location programmed or not.

NOTE

This function should not be used on any PROM which requires several passes thru the address range to program it, e.g., the 2708.

read_location

SYNTAX

read_location <address>

FUNCTION

This function will read the given location and display the data on the status line.

display

SYNTAX

display <ADDRESS>

Parameters

<ADDRESS> starting address of data display, default 0

FUNCTION

The display command will show the hex value of data stored at each location in the PROM.

Operating the Tape-Cartridge Based PROM Programmer

The following pages provide operating information for the user of the tape-cartridge based PROM Programmer. All of the information provided for the disc-based PROM Programmer applies to the tape-cartridge based PROM Programmer, except as noted in the following pages.

When using this PROM Programmer, the terminal will display a running history of the commands you entered so that you can see the programming configuration you have established.

File Data

The tape-cartridge based PROM Programmer writes to and reads from the emulation memory of the terminal if that memory has been configured by the emulator tape cartridge. This PROM Programmer can not read from or write to the user memory.

Absolute files can be created by the tape-cartridge based PROM Programmer, the emulation terminal, or the disc-based 64000 system. This PROM Programmer is not capable of either creating or using listing-type files.

Memory

The tape-cartridge based PROM Programmer writes to and reads from the emulation memory of the terminal if that memory has been configured by the emulator tape cartridge. This PROM Programmer can not read from or write to the user memory.

Operating Procedures

1. Set the address switches on the rear panel of the terminal to select LOCAL MASS STORAGE.

2. Install the PROM Programmer tape cartridge in the tape drive.

3. Press SHIFT and RESET together. This will begin a reboot of the terminal from the PROM Programmer tape cartridge. The display will show BOOT IN PROGRESS while the PROM Programmer routines are being loaded into the system.

4. The display will ask the question, "Do you want to run prom pv?" Make sure that you have a 64502A Programmer Module installed before you answer yes. Only the 64502A Programmer Module is capable of handling the voltage produced during Performance Verification.

   a. If you answer no , RETURN , the terminal will skip over the performance verification routines and go directly to the PROM programmer software.

   b. If you answer yes , RETURN , the display will be formatted to record the number of tests run and the number failed, if any. The STATUS line will prompt you to select the tests desired from those available on the softkeys. The system will display "Digital Test Passed" when the test is completed. This is the extent of the operator's self-check. For a detailed test, refer to the PROM Programmer Service Manual.

5. If the PROM programmer hardware in your terminal is capable of accepting and programming more than one PROM type, the command line will ask, "prom type?" The softkey label line will show the types of the PROM's which can be read and programmed by the hardware in your terminal. The status line will prompt you to enter the type of the PROM you will program. Press the softkey corresponding to the desired PROM number and press RETURN .

6. The display will show the type of PROM you selected along with the size of the PROM storage area and the address range of the PROM.

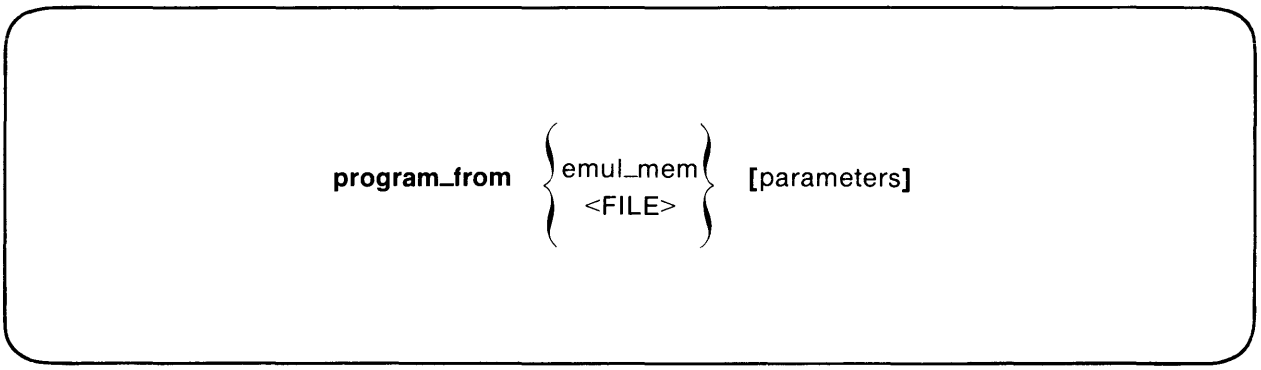
7. The STATUS line will prompt you to enter a command from those displayed on the softkey label line. Refer to table 11-3 for a quick reference to the function selected by each softkey. Each of the softkeys is described in detail on the pages following table 11-3.

8. Remove the PROM Programmer tape cartridge.

Table 11-3. Tape-Cartridge Based PROM Programmer Softkeys

Softkey Name	Function Selected
program	Causes the terminal to program and verify your PROM from an absolute file or from emulation memory.
verify	Causes the terminal to compare the data in your PROM with the data in an absolute file or emulation memory.
write_to	Causes the terminal to create a copy of the content of your PROM either in an absolute file or emulation memory.
check_sum	Causes the terminal to display a 32-bit total of all data stored either in emulation memory, in your PROM, or in the absolute file you specify.
prog_loc	Causes the terminal to program only the PROM address you specify.
end	Causes the terminal to exit any activity in progress and initiate a complete reboot of the system.
display	Causes the CRT to display the data at any PROM location you specify.

SYNTAX



Parameters

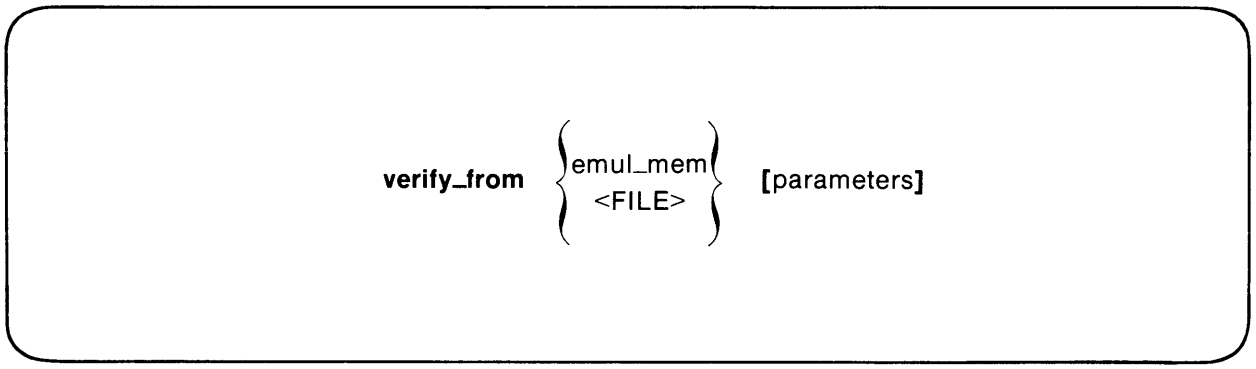
start<value>	starting file address, default 0
bit<value>	lsb of data unit, default entire PROM
rom_addr<value> thru<value>	PROM sub range, default entire PROM
neg_data	negative logic for data, default positive if not given
neg_addr	negative logic for address, default positive if not given
no_blank_check	turns off the automatic blank check feature

FUNCTION

The program command initiates the programming of a PROM from an absolute file residing on a tape cartridge or from emulation memory. Each time a PROM location is programmed, the location is read and compared with the data in the file. If the comparison does not match, the location is reprogrammed and compared again. Each location can be reprogrammed up to five times.

The PROM Programmer will make an automatic verification of the PROM. This is to ensure that the programming of one location did not affect the data in any other.

SYNTAX



Parameters

start<value>	starting file address, default 0
bit<value>	lsb of data unit, default 0
rom_addr<value> thru<value>	PROM sub range, default entire PROM
neg_data	negative logic for data, default positive if not given
neg_addr	negative logic for address, default positive if not given

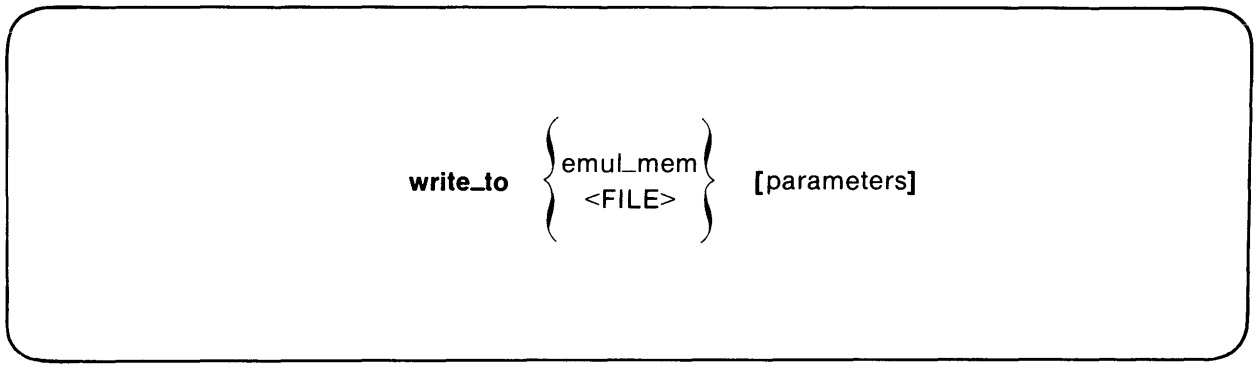
FUNCTION

The verify command will cause the terminal to compare absolute file data or emulation memory data with the data in the PROM and report the number of failures on the status line. If the PROM verifies, the status line will show:

Verify complete, no errors

If the PROM does not verify, the number of failures will be displayed along with an entry describing the first failure that was detected. If the verify does not find any data in the range specified, a warning message will be displayed stating that no locations were verified.

SYNTAX



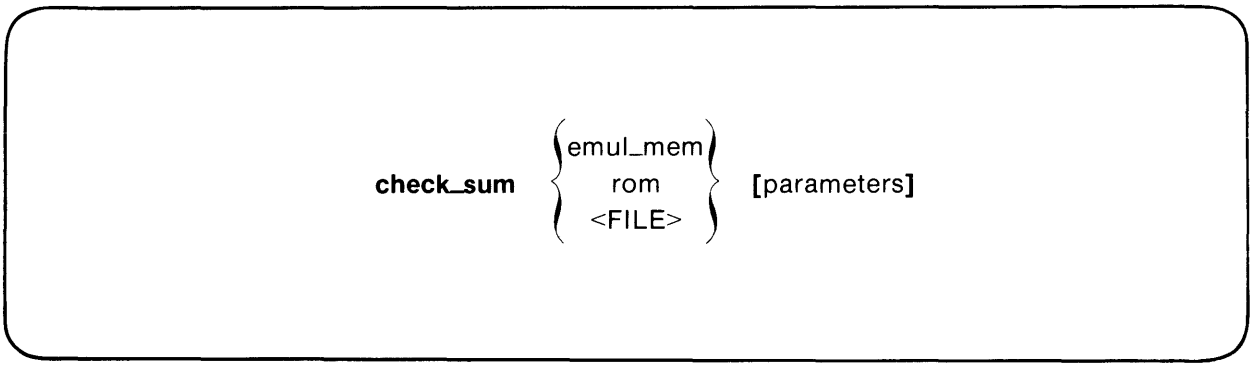
Parameters

init_tape	erases content of tape before writing new absolute file, default file written after last entry on tape.
start <value>	starting file address, default 0
bit <value>	lsb of data unit, default 0
rom_addr <value> thru <value>	PROM sub range, default entire PROM
neg_data	negative logic for data, default positive if not given
neg_addr	negative logic for address, default positive if not given

FUNCTION

The write_to command causes the PROM Programmer to create an absolute file of the PROM contents on tape, or write the PROM contents into emulation memory.

SYNTAX



Parameters

start<value>	starting file address, default 0
bit<value>	lsb of data unit, default 0
rom_addr<value> thru<value>	rom sub range, default entire rom
neg_data	negative logic for data, default positive if not given
neg_addr	negative logic for address, default positive if not given

FUNCTION

The check sum is a 32-bit total of all data in the specified range. The check_sum can be obtained for data stored in the PROM, the absolute file, or the emulation memory.

end

SYNTAX

```
end
```

FUNCTION

The end command will cause the terminal to halt any activity in progress and initiate a complete reboot of the operating system from any tape installed in the tape drive.

display

SYNTAX

```
display <ADDRESS>
```

Parameters

<ADDRESS>	starting address of data display, default 0

FUNCTION

The display command will show the hex value of data stored at each location in the PROM.

Error Messages

Table 11-4 lists all of the messages that may be displayed when you are using the PROM Programmer.

Table 11-4. PROM Programmer Error Messages

- Bit number too large

  Bit number specified is too large.

- Blank check failed

  PROM is not blank.

- Continue with test?

  This question only appears after the PROM Programmer has warned you of an improper hardware configuration. If you answer yes, the PROM Programmer will allow you to continue the test.

- Ending rom address too large

  Specified end address is greater than maximum address.

- Error found while reading listing file

  This message indicates that an error was found in the list file. The error was probably made while editing the list file.

- Install 64502A prom module

  The PROM Programmer self test has detected that there is no programmer module (PROM socket) attached to the control cable.

- Mapping error

  During programming, data was sent to an address specified as illegal in your memory map.

Table 11-4. PROM Programmer Error Messages (Cont'd)

- More than one prom card in cage

  Two or more cards in cage. There must be only one.

- No control card in card cage, power down and insert card

  Control card not plugged in. Turn off station power and start again.

- Programming error: prom address=XXXXH, data=XXH, file data=XXH

  The indicated location failed to program. Contents of the PROM are displayed along with data from the file which should have been programmed in that location.

- Prom module not recognized

  The socket is not recognized by the software. This message indicates that there is either a hardware error or that the software does not recognize a new socket.

- Socket module not connected

  Socket is not connected to PROM Control Card. Check cables.

- Starting rom address too large

  PROM address is greater than maximum address.

- Test voltage overload, check prom or socket

  When programmer is initialized, the socket is checked to see if it is drawing too much current. If this error message occurs, it indicates that there is either a hardware problem or that the PROM is plugged in upside down.

- User memory

  During programming, data was sent to an address specified as user memory in your memory map.

- Verify failed XXXX times

  This message indicates that the file data and PROM data did not match during the verify command.

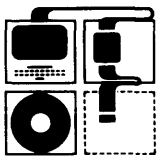
Table 11-4. PROM Programmer Error Messages (Cont'd)

- Warning, no file data found in address range.

  When verifying a file, if none of the file data is mapped into the PROM space, this message will be displayed.

- Warning, Test may damage prom module, Install 64502A

  This message is displayed when the PROM Programmer self test detects any PROM socket other than a 64502A Programmer Module installed. Only the 64502A Programmer Module is capable of handling the voltage produced during performance verification.



64500-90906

PRINTED IN U.S.A.