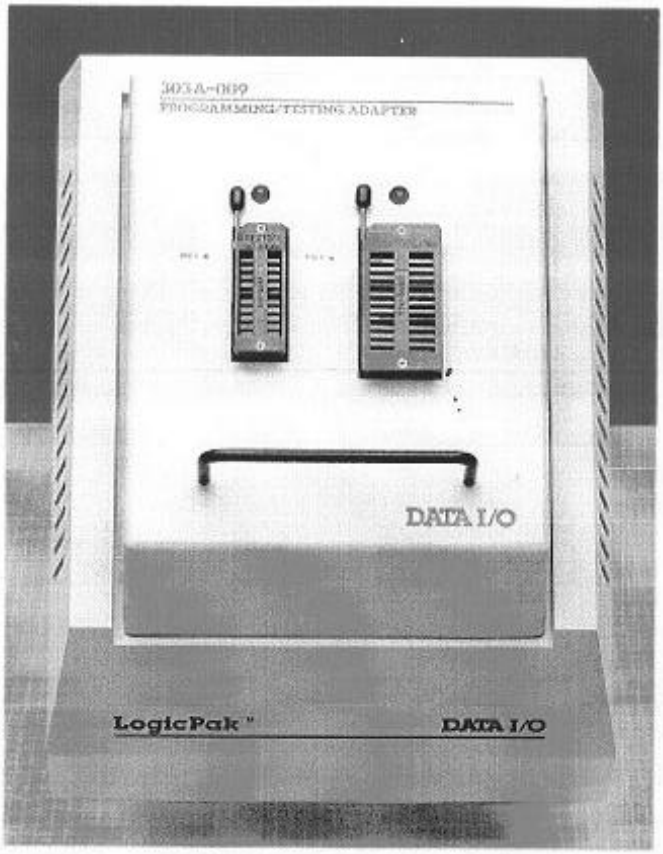
Programmable Logic Development System



Operator's Manual

DATA I/O

Data I/O has made every attempt to ensure that the information in this document is accurate and complete. However, Data I/O assumes no liability for errors, or for any damages that result from use of this document or the equipment that it accompanies.

Data I/O reserves the right to make changes to this document without notice at any time.

ORDERING INFORMATION

When ordering this manual use part Number 981-0142-005.
Applies to Engineering Part Number 950-1942-008.

SAFETY SUMMARY

General safety information for operating personnel is contained in this summary. In addition, specific WARNINGS and CAUTIONS appear throughout this manual where they apply and are not included in this summary.

Definitions

WARNINGS statements identify conditions or practices that could result in personal injury or loss of life.

CAUTION statements identify conditions or practices that could result in damage to equipment or other property.

Symbols

⚠ : This symbol appears on the equipment and indicates that the user should consult the appropriate manual for further detail.

∿ : This symbol stands for Vac. For example, 120V∿ = 120 Vac)

Power Source

Check the voltage selector indicator (located inside the rear panel) to verify that the product is configured for the appropriate line voltage.

Grounding the Product

The product is grounded through the grounding conductor of the power cord. To avoid electric shock, plug the power cord into a properly wired and grounded receptacle only. Grounding this equipment is essential for its safe operation.

Power Cord

Use only the power cord specified for your equipment.

Fuse Replacement

For continued protection against the possibility of fire, replace the fuse only with a fuse of the specified voltage, current and type ratings.

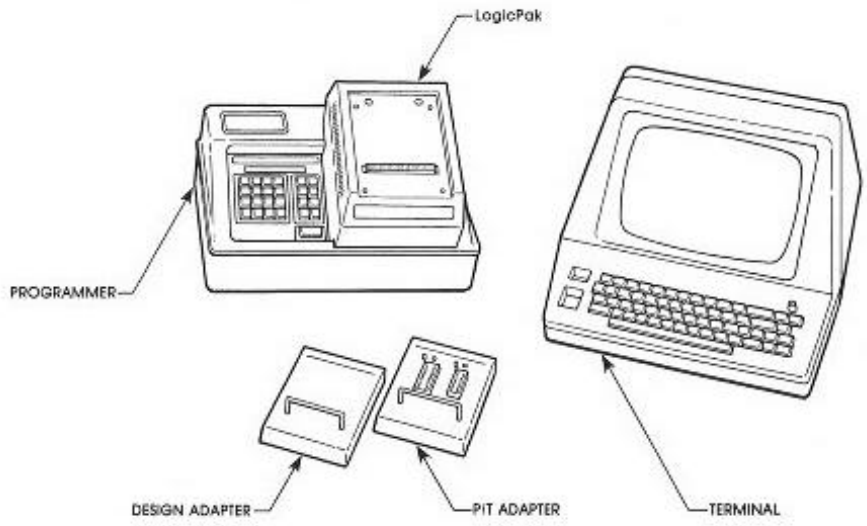
Servicing

To reduce electric shock, do not perform any servicing other than that described in this manual.

TABLE OF CONTENTS

INTRODUCTION

Data I/O's Programmable Logic Development System (PLDS) provides data development, programming, and testing support for logic device families from most semiconductor manufacturers. This modular system comprises a programmer, the LogicPak™, adapters, and a terminal as shown in the accompanying figure.



This manual contains the operational procedures of the LogicPak as well as the necessary operational information of the other components of the PLDS. Included in this manual are instructions on:

- ☐ GETTING STARTED—A sample session. Includes instructions on powering up the programmer, installing the LogicPak, installing an adapter, inserting a device, and programming the device.
- ☐ PROGRAMMING—Information about programming devices, including a list of general programming notes. Also includes information on editing, verifying data integrity, and serial port data transfer.
- ☐ REMOTE CONTROL—Provides descriptions of the remote control options available.
- ☐ SYSTEM COMMANDS—Details the select codes available from the programmer keyboard and the terminal.
- ☐ INDEX—An alphabetical guide to all the major topics covered in the manual.
- ☐ APPENDICES—A collection of reference material.
  - A. Error Codes—Describes the PLDS error code displays, and corrective action.
  - B. Test Modes—Provides description and operating information on the test options of the LogicPak.
  - C. JEDEC Format—Provides an overview of the JEDEC format for the transfer of information between a data preparation system and a logic device programmer.

INTRODUCTION

Features

The PLDS programs a wide variety of logic devices, with a minimum of equipment changes. Because P/T (program/test) adapters configure the LogicPak's hardware/firmware to specific manufacturers' device families, the only hardware you have to change when you change device families is the P/T adapter.

Data I/O offers three methods for data development. Data can be developed using ABEL™, a programmable logic design tool for PAL™ and IFL devices. ABEL (Advanced Boolean Expression Language) allows you to describe the desired operation of both PAL and IFL devices with Boolean equations, Boolean sets, logic function tables, or state diagram entry methods.

Two design adapters are available: the PALASM™ design adapter for PAL's and H & L design adapter for IFL devices.

The LogicPak can accept JEDEC file downloads (see Appendix C) from other development systems, such as ABEL.

Functional testing of logic devices is necessary to detect a functional failure in a device which could pass a routine fuse-verify test. There are two methods of functional testing available using the LogicPak.

A structured vector test may be performed on the device automatically after programming if there are structured test vectors present in the programmer data RAM. Programmed devices can be functionally tested using an optional method developed by Data I/O called the Logic Fingerprint test. This test finds defective devices using a signature analysis technique.

System Overview

The Data I/O 303A LogicPak contains all the common electronics and firmware for the programming and testing of logic devices. Any electronics or firmware unique to a specific-device family, or device within a family, are resident in P/T adapters that plug into the LogicPak. Families with more than one pin number series (eg., PAL 20 and PAL 24) have sockets to accommodate each package size. Device families that are available in different packages, such as 20-pin DIP, 24-pin DIP, 28-pin DIP, as well as leaded and leadless chip carriers, are accommodated by the P/T adapters.

To increase flexibility in waveform generation, digital- to- analog converters (DAC) control all major power supplies, with several rise and fall times selected by software.

System Capabilities and Operational Summary

The LogicPak can be used with the Model 29 Universal Programmer, the System 19, or 100A programmers (see the subsection on programmer compatibility).

To program a logic device, you must first load the desired state of the device fuses into the programmer RAM. Fuse information can be input in several forms: JEDEC file information developed with ABEL, decimal fuse number and state, H & L programming tables for IFL devices, and Boolean equations for PAL devices (PALASM). The detailed operating procedures for data development using ABEL or the design adapters are provided in the ABEL, PALASM and H & L manuals.

Once the fuse pattern is loaded into RAM, you can program the device using a P/T adapter. Thus, the design adapter can be removed from the LogicPak, and an appropriate P/T adapter can be installed allowing the fuse pattern to be programmed directly into the device and automatically tested. The programming steps are virtually the same as those taken when programming PROMs. To start the programming sequence, first enter a device code, then programming can begin (see the Getting Started section for a sample programming procedure and your programmer manual for further operating details).

Because a programmable logic device is not completely manufactured until it is programmed, the programmer must be able to perform some type of functional testing. This is accomplished by using structured vector testing and/or Data I/O Logic Fingerprint testing.

Specifying a Fusemap

The design adapter firmware translates your design into a fuse pattern and, optionally, test vectors. This fuse pattern and test data are resident in the programmer's RAM. If a fuse pattern is generated on a host system, it must use fuse numbers specified according to logic diagrams in each P/T adapter User Note and transmitted to the programmer in the JEDEC (Joint Electron Device Engineering Council) format (see Appendix C). Data I/O uses the JEDEC Logic Device Translation Format (number JC-42, 162) for serial data input and output with the LogicPak. The only exception to this is when you are using a Signetics H & L design adapter, in which case data transfer can also occur in the Signetics H & L logic format.

An alternate method of specifying the fusemap is to manually enter the fuse number and state (see the logic diagrams for each adapter in the user notes for the particular adapter) for every fuse in the device. These diagrams are the same as those in the device manufacturers' data books, but the fuse numbers have been added. Although the task is tedious, fuse numbers and states can be entered manually into the programmer's data RAM from the programmer's keyboard or from a terminal using a fuse editor. This method usually will be used only for editing fuse data because it is a long process with room for error.

The P/T Adapter

With a P/T adapter, fuse data can also be entered into the programmer's RAM by loading from a master device. Blank devices can then be programmed using the same P/T adapter, or other manufacturers' functionally equivalent second-source devices can be programmed by installing the appropriate P/T adapter. Remember that a device that has its security fuse programmed cannot be used as a master because its fuses cannot be read.

A different programmer RAM fusemap was used in some previous Data I/O logic device programming modules (950-0800, 919-1427, and 919-1542). If you have paper tapes or files that you prepared to use with these modules, you must prepare new files for use with the LogicPak unless you used the Signetics H&L logic format. The preferred translator format is the JEDEC format, but a Signetics translator is available. (The 950-0104 and 919-0045 modules use a memory map that is compatible with the LogicPak.) The 919-1427 pak used a fusemap generated to simulate a 512 x 4 PROM. Serial data were entered using a standard PROM data translation format. Again, tapes must be regenerated for use with the LogicPak because of the different correspondence between fuses and bits in RAM. The fusemaps have been changed to allow for the programming of functional second-source devices from the same fusemap loaded in RAM and to avoid gaps (previously called phantom fuses) in the fusemaps.

Sumcheck

After fuse data have been loaded into the programmer from the serial port or the master device, the programmer calculates the sumcheck of the fuse data (see the next figure) and displays it (when in terminal mode, it will be displayed on the terminal). The sumcheck, which is used to verify the integrity of data transfers, is a summation of eight-bit bytes of fuse data expressed as a four-digit hexadecimal number as shown in the table.

Hexadecimal Data	Binary Data
84	10000100
C1	11000001
62	01100010
24	00100100
01CB	0000 0001 1100 1011
16-Bit sumcheck in hexadecimal notation	16-bit binary hexadecimal sumcheck

If data was loaded through the serial port and a sumcheck was sent with it, the programmer will compare the sumcheck with its own calculation. If they agree, the correct sumcheck is displayed. If they do not agree, the programmer will signal an error. Data from the serial port will also be checked for correct parity if the programmer parity switch is on (see your programmer manual).

Data from a master device are loaded into RAM by installing the correct P/T adapter on the LogicPak and performing a load operation. Any information in the source buffer (Boolean equations, function tables, or test vectors) will not be affected. The source buffer is used in conjunction with the design adapters as a holding area for the translation of the Boolean equations into the device fuse map.

Test Capabilities

With the LogicPak installed on the programmer, when power is applied, the programmer will perform a series of self-tests to ensure functionality. A failure will display an error code (see Appendix A). To isolate the problem to its source, refer to the maintenance manual.

Functional Testing verifies that a programmed device will perform as intended. Descriptions of how to test devices is in the System Commands section. Additional information is given in Appendix B.

- ☐ Fuse Verify—A test to verify that the fuse pattern in the device and the programmer RAM are the same. This test is run automatically during the programming cycle.
- ☐ Structured Test—When test vectors are entered, the structured tests are always performed prior to the Logic Fingerprint test. If no test vectors are present, only a fuse verify and a Logic Fingerprint test (if enabled) will be performed.
- ☐ Logic Fingerprint Test—This is the easiest test to use and is flexible enough to be applicable to a wide variety of logic devices. Refer to your P/T adapter User Note for Logic Fingerprint Test limitations.

Programmer Compatibility

To be compatible with the LogicPak, your programmer may require a hardware and/or firmware update, depending on the model, configuration, and age. The information that follows will help you determine whether your programmer requires updating. If you find that your programmer does require updating, contact your nearest Data I/O customer support center. (A list of customer support centers is at the back of this manual.)

The 29B programmer has no special compatibility requirements for proper operation with the LogicPak. Refer to your P/T adapter User Notes for any additional compatibility requirements that may be necessary.

- ☐ System 17—The System 17 must be converted into a System 19 with the latest firmware installed and latest hardware modifications.
- ☐ System 19—Check to determine whether your System 19 contains a 702-1520 or 702-1980 controller board by performing the following steps:

  1. Remove the programming pak.
  2. Remove the metal or plastic shield (if any).
  3. Count the number of EPROM firmware sockets located just behind the pak interface connector. If there are four sockets, it is a 702-1520 board. If there are eight sockets, it is a 702-1980 board.

  If your System 19 contains a 702-1520 controller board, check the modification status sticker on the bottom of the programmer. If the sticker is not there, or if the "1" is marked off, your System 19 requires hardware and firmware updating; contact the nearest Data I/O customer support center. If "2" is marked, your System 19 is compatible with the LogicPak. If your System 19 contains a 702-1980 controller board, it may require a firmware update. To display the configuration number of the firmware in your programmer, key in "SELECT-B2-START" and observe the display.

  If the configuration number displayed is "3599" or "CC8B," your firmware needs updating.

- ☐ Model 29A Universal Programmer—To be compatible with the LogicPak, the Model 29A programmers must have Rev (revision) C or later firmware. To determine the configuration of the firmware in your Model 29A, key in "SELECT-B2-START" and observe the display. If the hexadecimal number matches one listed in the following table, your firmware needs to be updated.

Model 29A and 100A Programmers Requiring a Firmware Update

Model	Rev	Configuration Number
29A	A	1ECA
	B	20A4
29A (with computer remote control)	A	BB41
	B	C00B
100A	A	917F
	B	9405
	C	9DEE
	D	9BED

- ☐ 100A Production Programmer—To be compatible with the LogicPak, the 100A programmers must have Rev E or later firmware. To determine the configuration of the firmware in your 100A, key in "SELECT-10-START" and observe the display. If the hexadecimal number display matches one listed in the table, your firmware needs to be updated.

Specifications

The physical and environmental specifications of the LogicPak are:

- ☐ altitude (operating): sea level to 3 km (10,000 ft)
- ☐ humidity (operating): 90% maximum (noncondensing)
- ☐ humidity (storage): 95% maximum (noncondensing)
- ☐ temperature (operating): 5 to 45 °C (41 to 113 °F)
- ☐ temperature (storage): -40 to 70 °C (-40 to 158 °F)
- ☐ weight: 1.6 kg (3 lb, 8 oz)
- ☐ dimensions: 17.9 x 17.3 x 21.7 cm (7.05 x 6.81 x 8.54 in.)

Applications

As Data I/O increases the capabilities of the LogicPak to program new or additional devices, firmware updates will be available for existing adapters to add new devices to existing-device families. New adapters may also be added to the LogicPak to accommodate new device families.

Warranty and Customer Support

Data I/O equipment is warranted against defects in materials and workmanship. The warranty period of one year, unless specified otherwise, begins when you receive the equipment. Refer to the warranty and inside the back cover of this manual for information on the length and conditions of the warranty. For warranty service, contact your nearest Data I/O Customer Support Center.

Data I/O maintains customer support centers throughout the world, each staffed with factory-trained technicians to provide prompt, quality service. This includes not only repairs, but also calibration of all Data I/O products. A list of all Data I/O customer support centers is located in the back of this manual.

Ordering

Orders made with Data I/O must contain the following information:

- ☐ Description of the equipment
- ☐ Quantity of each item ordered
- ☐ Shipping and billing address of firm, including ZIP code
- ☐ Name of person ordering equipment
- ☐ Purchase order number
- ☐ Desired method of shipment

Options

The following items can be ordered by contacting your Data I/O service representative.

Options	
303A-001	IFL P/T Adapter
303A-002	MMI/National PAL P/T Adapter
303A-003	Harris P/T Adapter
303A-004	AMD PAL P/T Adapter
303A-006	Texas Instruments P/T Adapter
303A-007	Harris CMOS P/T Adapter
303A-008A	32R16 P/T Adapter (DIP + LCC)
303A-008B	32R16 P/T Adapter (DIP + NLCC)
303A-009	CMOS P/T Adapter
303A-010	Altera/Intel 40-Pin P/T Adapter
303A-100	PALASM Design Adapter
303A-101	Signetics H & L Design Adapter

GETTING STARTED

This section explains how to get started using your LogicPak and a Model 29 programmer. Refer to the programmer manual for the procedure to perform a load operation. This section also provides the necessary procedures for connecting the other components of the PLDS. Included here are complete procedures for powering up and for programming a device from a master device and using your programmer keyboard. For details on operating your programmer, refer to your programmer manual and the sections on Programming and Remote Control of this manual.

This section includes the following information:

- ☐ Installing the LogicPak on the Programmer
- ☐ Installing a P/T Adapter on the LogicPak
- ☐ Sample Programming Session

Power Connection

Since the LogicPak is a plug-in module that is inserted on your programmer, the power connection procedures you will follow are those for your programmer. The LogicPak alone has no power connections. Refer to your programmer manual for the procedures.

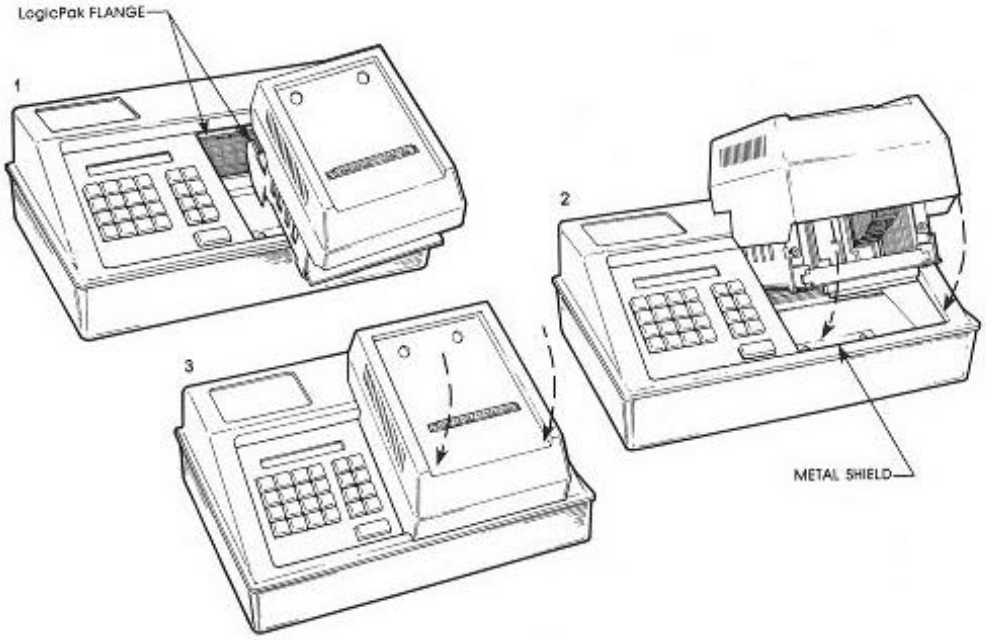
LogicPak Installation

The LogicPak may be installed and removed with the programmer's power on; this feature allows you to retain data in RAM during module changes. If the programmer power is turned on before the LogicPak is installed, you will hear a beep until the LogicPak with an adapter is installed.

CAUTION

Voltage transients can cause device damage. If a P/T adapter is installed in the LogicPak, be sure that all sockets are empty when switching power on or off, before installing or removing the LogicPak or adapter.

To install the LogicPak into a programmer, refer to the figure and follow this installation procedure.



NOTE

Although the Model 29 is shown here, the insertion procedure is the same for all systems.

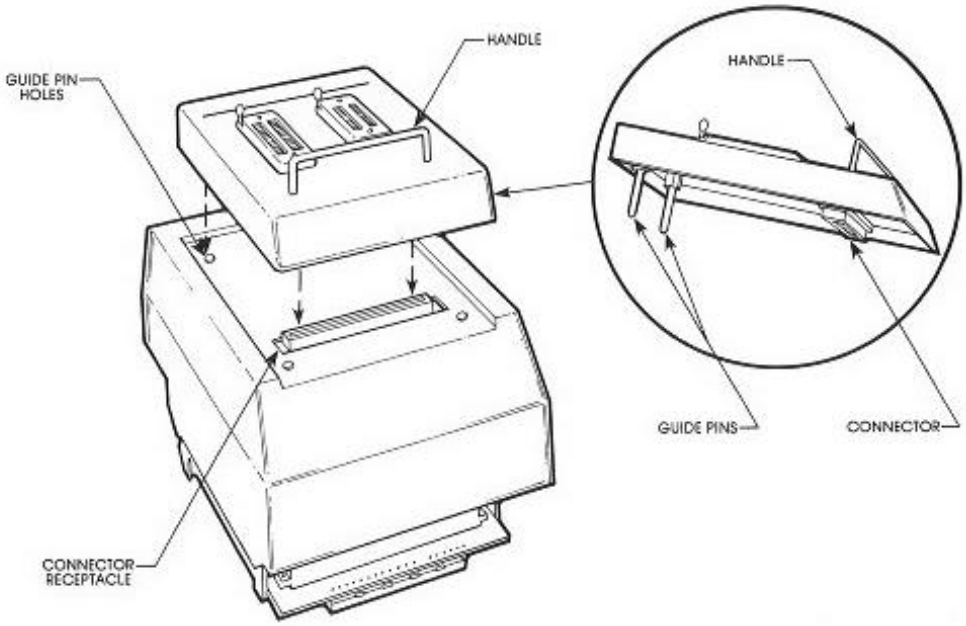
1. Slide the LogicPak into the opening in the programmer.
2. Tilt the LogicPak up, and gently push it back to hook its flange over the back edge of the programmer opening.
3. Lower the LogicPak into position as shown in the figure.
4. Press down gently on the front of the LogicPak to ensure a good connection.

CAUTION

Be careful when inserting the LogicPak. Always hold the LogicPak by the chassis and not by the adapter handle when carrying it.

Adapter Installation

To insert all adapters into the LogicPak, refer to the figure and follow this installation procedure.

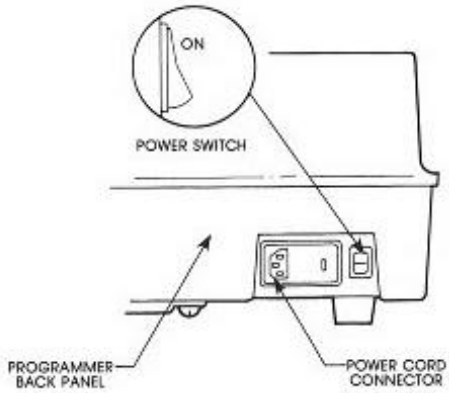


1. Check to make sure a device is not in a socket. If a device is in a socket, remove it by lifting the lever on the side of the socket and then lift the device out of the socket.
2. Align the guide pins on the underside of the adapter with the guide pin holes on the LogicPak as shown in the figure.
3. Gently set the adapter on the LogicPak.
4. Firmly press down on the front edge of the adapter to lock the connector pins into the connector receptacle.

Powering Up

The first step in getting started is powering up your programmer. Use the following procedure.

1. Check to make sure the adapter sockets are empty. If a device is in a socket, remove it.
2. Check to be sure the voltage selector is in the proper position. Plug the AC power cord into the rear of the programmer, and into a power receptacle.
3. Press the power switch at the back of the programmer to the "ON" position as shown in the figure below.



When the programmer is powered up, it automatically performs a self-test routine. This will take a few moments and the programmer will display indications that the test is being performed.

Sample Programming Session

The following steps describe how to program a 16R8 device using a master device (a part that has been previously programmed and is used as a "master" to program other parts). This procedure assumes that the LogicPak is installed in the programmer. The programming section in your programmer manual gives complete descriptions of the commands and procedure.

NOTE

This programming session assumes operation and key entry from the programmer front panel. These programming operations can also be executed from the terminal and are discussed in the System Commands section of this manual.

1. Press [COPY] [DEVICE] [RAM] [START]

   to prepare the programmer to transfer the fuse pattern data from the master device to the programmer's RAM. Refer to your P/T adapter manual or User Note for complete listings of family/pinout codes for each adapter. Find the code numbers corresponding to the device number for the manufacturer of the device. These codes are found under the column "family/pinout". The programmer will display

   FAM 00 PIN 00

2. Press [2] [2] [2] [4]

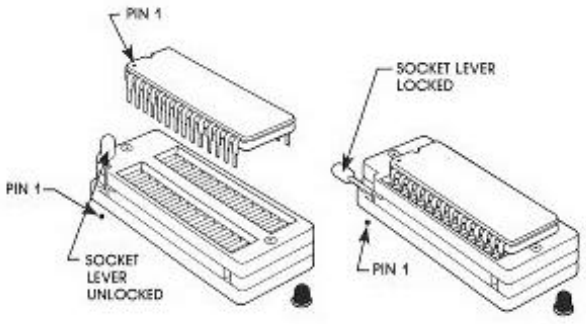
   the family/pinout code for the MMI 16R8 device. The programmer will then display

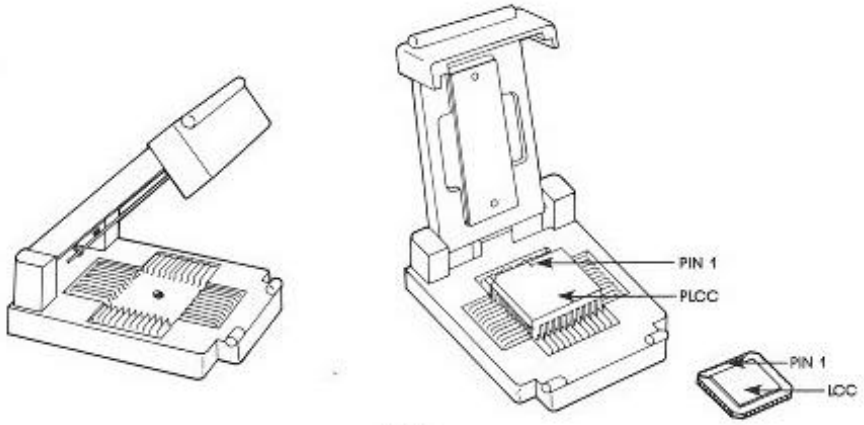
   FAM 22 PIN 24

NOTE

The 303A-008 P/T adapter only programs the 32R16 PAL; therefore the keyboard or terminal will always display "2247", the family and pinout code for the 32R16 PAL. These codes are set automatically when power is turned on. No other codes will be accepted.

3. Lift up the lever on the socket that has an illuminated LED below it (see figure). Line up pin 1 of the device so that it is nearest the pin 1 indication dot and set the device into the socket. Press down on the lever to lock the device in place.





NOTE
For LCC devices, orient pin 1 according to the drawing in your P/T adapter User Note.

4. Press START . The programmer will display

   LOADING DEVICE
   LOAD DONE XXXX

NOTE
XXXX is the sumcheck of the device. See step 8 for more information.

5. Lift up the socket lever and remove the master device from the socket. The master device fuse data is now transferred to RAM. The next part of the procedure transfers that fuse data to the blank device.

6. Press COPY RAM DEVICE START
   to prepare the programmer to transfer the fuse data from RAM to the blank device. The programmer will display

   FAM 22 PIN 24

7. Line up pin 1 of the blank device so that it is nearest the pin 1 indication dot and set the device into the socket. Press down on the lever to lock the device in place.

8. Press START . The programmer will display

   TEST DEVICE
   PROGRAM DEVICE
   VERIFY DEVICE
   PRG DONE 01 XXXX

NOTE
XXXX in the above display represents the device's sumcheck, the hexadecimal sum of all the bytes in the device. The number displayed should match the sumcheck displayed during step 4 of this procedure. "PROG DONE 01" means that 1 device has been programmed.

9. Lift up the socket lever and remove the device from the socket. The device is now programmed.
10. To program another device, simply place it in the socket and press START.

PROGRAMMING

After fuse data development, the next step in the programming sequence is programming the logic devices. The LogicPak applies the manufacturer's specific algorithms to blow fuses in the logic device according to the fuse pattern data in the programmer RAM. Once you key in the operation on the programmer, programming is automatic and starts with a series of tests: backward device test, illegal-bit test, and blank check. During the backward device test, the programmer automatically checks the device's orientation in the P/T adapter socket and displays an error if it is inserted backwards. The Model 29 displays:

DEV BACKWARDS 32

The illegal-bit test checks for previously programmed bits in a nonblank device that should not be programmed according to the fuse pattern in RAM. If illegal bits exist, the Model 29 displays:

ILLEGAL BIT 21

During the blank check, the programmer searches the device for blown fuses. If any are found (and the bits are legal), the programmer will signal the operator and the Model 29 displays:

NONBLANK 20

Nonblank parts can be over-programmed by again pressing

START

If the device passes these tests, data are transferred from the programmer RAM to the LogicPak. The LogicPak then applies the programming pulses to the appropriate pins and tests the state of the selected fuse condition. If the fuse fails to program, the Model 29 displays:

PROGRAM FAIL 22

Otherwise, programming proceeds to the next fuse until all have been programmed. (With some P/T adapters, programming algorithms vary and may not display the "program fail" error message. These adapters will, however, display a "verify fail" message if a fuse fails to program.)

The Model 19 will display the following error codes:

ERR 32
ERR 22
ERR 21

A blinking display of the current RAM address and data implies the device is nonblank

Refer to Appendix A for a listing of the error codes generated by the LogicPak. If an error code does not appear there, check the programmer manual.

Editing Features

After data have been developed or loaded into programmer RAM, the PLDS provides editing features that allow you to modify the fuse pattern. These are accessed through one and two-digit commands entered on the programmer keyboard or on a terminal. Refer to the Edit Fuse Pattern subsection in the System Commands section for more details.

REMOTE CONTROL

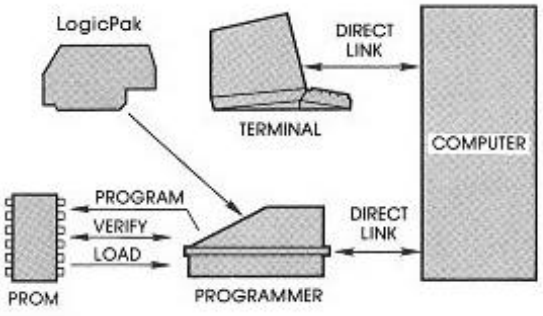
This section of the manual provides descriptions of the remote control options available for programmer/LogicPak operation.

The LogicPak uses the RS-232C serial interface of the programmer for interaction with a terminal and for communication with a host computer. For complete interconnection methods see your programmer manual.

Computer Remote Control

Computer Remote Control (CRC) is designed to enable you to control the programmer by a computer. Linked directly to the programmer, the computer generates and sends commands to the programmer, determines variables for setting programming parameters (where needed), and reacts to information, returned to it from the programmer. While these commands may be sent by an operator at a terminal, the commands and syntax were designed to be easily incorporated into a computer program.

The figure below illustrates the basic components of a logic development system under computer remote control. For interactive programs, the computer can send messages to be displayed on the programmer, and can read keystrokes from the programmer's keyboard (29B V03 or later only). The user must provide application software which will allow the computer to issue CRC commands and to interpret CRC responses. The list of commands and responses is contained in the programmer manual.



For device programming convenience, you can also control your Data I/O programmer from an IBM PC with Data I/O's menu-driven remote control software package PROMlink. Data files can be stored and retrieved from the PC, eliminating the need for master devices or paper tapes. Complete control of the programmer is also accomplished by this method.

Terminal Remote Control

System Command E1 transfers control of the programmer to the terminal. After control is transferred, the programmer will display only its action symbol. The E1 command allows you to menu driven access to data development and remote operations resident in the design adapters and remote operations using P/T adapters.

SYSTEM COMMANDS

The LogicPak offers numerous system commands that allow you to edit and transfer data and set parameters. System commands are accessed by entering a two-character select code from the programmer front panel or a menu indicated code from the terminal. Some commands will prompt for data entry. The system commands and a brief description are listed in the Command Summary table.

The sequence explanations assume no operating errors. If these occur, the programmer signals with a beep and displays a two-digit error code in front panel mode or an error message in terminal mode. It also beeps once when an incorrect key is pressed. Error codes are explained in Appendix A and in your programmer manual. Some errors will return you to the programmer front panel control from the terminal mode.

The entries that you are to make from either the programmer or the terminal are indicated by the entry enclosed in a key symbol. For example:

indicates the ESC (escape) key on the terminal keyboard should be pressed.

When the entry you are to make is variable, appropriate substitutions for the actual values will be used, for example:

indicates that the appropriate values for the family and pinout code for the device will be entered.

Command Summary

The following definitions describe the display options in the terminal mode. These same function operations are available from the front panel but there will be no terminal display.

Code	Command Name	Front Panel	Terminal	Description of Terminal Mode
E1	Enable Terminal Mode	X		Transfers control of the LogicPak and programmer to the terminal
0	Display Command Menu		X	Causes the LogicPak to redisplay its command menu on the terminal
1	Family and Pinout Code		X	Allows the user to enter the family/pinout code for a logic device
E5	Reject Count Option	X		Allows the user to select manufacturers recommended number of pulses or 1 pulse programming
5			X	
E6	Verify Option	X		Displays a three item menu for selecting verify and functional test routines
6	Option		X	
E7	Security Fuse	X		Displays and allows selecting of the security fuse options
7	Option		X	
E8	Functional Test	X		Enables the functional test data function. Allows you to enter functional test data and perform vector editing.
E9	Data	X		
8			X	
EA	Transmit Fuse	X		Transmits the fuse pattern in the programmer RAM to the serial port
A	Pattern		X	
EC	Transmit JEDEC	X		Transmits the contents of the fuse and vector RAM to the serial port in the JEDEC format (see Appendix C)
C	Data		X	
EB	Receive JEDEC	X		Prepares the programmer to receive JEDEC fuse and vector data from a peripheral device via the serial port
B	Data		X	

(continued on next page)

Code	Command Name	Front Panel	Terminal	Description of Terminal Mode
ED D	Display Fuse Sumcheck	X	 X	Displays the sumcheck of the fuse data in RAM
EE E	Edit Fuse Pattern	X	 X	Enables the fuse editing function. Fuse states may be changed from blown to unblown or vice-versa in fuse memory.
EF F	Display Config Number	X	 X	Displays the configuration number of the adapter firmware
CE G	Select Attributes	X	 X	Provides a menu of six attributes, each of which allows you to select one of two options
CTRL Z ESC	Exit Commands		X X	Allows you to exit a programming session and terminate all operations
CTRL Y	Abort Operation		X	Allows you to abort an operation such as fuse map dump, or structured vector error dump
2	Load RAM		X	Loads the programmer with data from a master device
4	Program RAM		X	Programs, verifies and tests a device with programmer RAM data
3	Verify Device		X	Verifies and tests a device

NOTE

These last three programming functions listed in the table are also available from the front panel. The key sequences for front panel operation are in the Getting Started section of this manual and in your programmer manual.

Load RAM with Master Device Data

Before loading the programmer with data from a master device:

1. Place the system in terminal mode by selecting E1 from the front panel of the programmer. The programmer should be connected to the RS-232C serial port of the terminal.
2. Enter the family and pinout code (refer to your P/T adapter User Note) at the terminal, if prompted by the terminal display.

NOTE

If options are desired such as performing a functional test, select options and parameters as needed before proceeding.

Use the following procedure to load the programmer with data from a master device from front panel mode.

Procedure	Front Panel Key Sequence	Front Panel Display
1. Prepare programmer to transfer transfer fuse pattern data from master device to programmer RAM.	COPY DEVICE RAM START	*FAM 00 PIN 00*
2. Accept (or if display is wrong), key in the 4-digit family/pinout code for the device (check the device list in your P/T adapter User Note). (X is the number entered.)	F F P P	*FAM XX PIN ^ XX*
3. Insert the master device into the socket with the illuminated LED below it.	START	*LOADING DEVICE* ⍁ *LOAD DONE XXXX* sumcheck of data transferred
4. Remove the master device		

An action symbol ⍁ will be displayed showing the pretesting, programming and verifying of the device. If no errors occur, the terminal displays sumcheck XXXX.

Use the following procedure to load the programmer with data from a master device from the terminal.

Procedure	Terminal Key Sequence	Terminal Display
1. Select the load operation.	2	Command : 2 - Load device Push return to load device
2. Insert the master device into the socket.		Command : 2 - Load device Push return to load device Loading device ... Suncheck 0000 Command :

An action symbol . . . . will be displayed showing the pretesting, programming and verifying of the device. If no errors occur, the terminal displays sumcheck XXXX.

Program Device with RAM Data

Before programming a device with fuse pattern data previously loaded into the programmer's RAM:

1. Place the system in the terminal mode (select E1 from the front panel).
2. Enter the family and pinout code, if prompted by the terminal.

NOTE

If options are desired such as programming a security fuse, select options and parameters as needed before proceeding.

Use the following procedure the program the device with data in RAM from the front panel mode.

Procedure	Front Panel Key Sequence	Front Panel Display
1. Prepare programmer to transfer fuse data to the blank device.	COPY RAM DEVICE START	*FAM 00 PIN 00*
2. Insert the blank device.	START	*TEST DEVICE* ☐ *PROGRAM DEVICE* ☐ *VERIFY DEVICE* ☐ *PRG DONE 01 XXXX*
NOTE *This step, along with programming the device, verifies that the data was correctly transfered.*		
3. Remove the device. To program another, insert a blank device and press	START	

An action symbol ☐ will be displayed showing the pretesting, programming and verifying of the device. If no errors occur, the terminal displays sumcheck XXXX.

Use the following procedure to load the programmer with data from a master device from the terminal.

Procedure	Terminal Key Sequence	Terminal Display
1. Select the program operation operation.	4	Command : 4 - Program device Push return to program device
2. Insert the blank device into the socket.	RETURN	Command : 4 - Program device Push return to program device Testing device .... Programming device .... Verifying device ..... Suncheck 0808 Command :

An action symbol . . . . will be displayed showing the pretesting, programming and verifying of the device. If no errors occur, the terminal displays sumcheck XXXX.

Verify and Functionally Test Device

Before verifying and testing a programmed device from terminal control:

1. Place the system in the terminal mode (select E1 from the front panel).
2. Enter the family and pinout code, if prompted by the terminal.

NOTE

If options are desired such as performing a functional test, specifying the number of passes, or choosing the verify modes, select options and parameters as needed before proceeding.

Procedure	Terminal Key Sequence	Terminal Display
1. Insert the device to be verified verified.	3	Command : 3 - Verify device Verifying device ..... Suncheck 0000 Command :

An action symbol . . . . will be displayed showing the verification function under way. Upon completion the terminal will display sum-check XXXX of the device fuses.

Enable Terminal Mode

With the programmer connected to the programmer RS-232C port of the terminal, the terminal will prompt you to enter family codes and pinout codes unless they have already been entered. The terminal will then display the command menu.

Procedure	Front Panel Key Sequence	Front Panel Display
1. Select the terminal mode.	SELECT E 1 START	

Display Command Menu

This command causes the PLDS to redisplay its command menu on the terminal, as shown in the previous figure.

Procedure	Terminal Key Sequence	Terminal Display
1. Display command menu on screen.	0	See the following display.

```
Command : 0 - Display menu

DATA I/O CORP. - Programmable Logic Development System - 303A-V04
                 Copyright 1982, 1983, 1984

     - GENERAL COMMANDS -                    - I/O COMMANDS -
0 - Display menu                        B - Receive JEDEC data
1 - Enter Family/pinout code            C - Transmit JEDEC data
5 - Enter reject count option
6 - Enter verify option
7 - Enter security fuse option
8 - Enter functional test data
F - Configuration number
G - Select attributes

   - DEVICE RELATED  COMMANDS -          - FUSE MAP COMMANDS -
2 - Load device                         A - Display fuse pattern
3 - Verify device                       D - Display fuse suncheck
4 - Program device                      E - Edit fuse pattern

NOTE - Always transmit an "ESC" before removing adapter
```

Family and Pinout Code

From the programmer front panel control, family and pinout code entry is part of device-related operations (see the Getting Started section in this manual and refer to your programmer manual). This procedure is for operation from the terminal.

NOTE

In the Family and Pinout Code list, find the code numbers corresponding to the device number for the manufacturer of the device. Notice that some devices, such as the AmPAL22V10, have two family/pinout codes. See the P/T adapter user notes for an explanation of the difference.

Procedure	Terminal Key Sequence	Terminal Display
1. Select the option that allows you to enter the family/pinout code.	[1]	Command : 1 - Enter Family/pinout code Family/pinout code 2224
2. Accept, or if display is incorrect key in the 4-digit family/pinout code for the device you are using.	[F] [F] [P] [P]	NA

NOTE

Space and backspace (CTRL H) may be used to move the cursor back and forth.

Set Reject Count Option

This command allows you to select the number of programming pulses applied to the device fuses before the programmer rejects the device as unprogrammable. The default value of 0 selects the manufacturer's specified number of programming pulses while a value of 1 selects a single programming pulse. Refer to the adapter User Notes for specific entries to select optional reject values.

NOTE

Altera, Cypress, and VTI devices are not supported by this option because of the type of programming algorithms used. The design adapters also do not provide this option.

Use the following procedure to select the reject count option from front panel mode.

Procedure	Front Panel Key Sequence	Front Panel Display	
1. Select the reject count option.	SELECT, 2, 5, START	*0 0*	
2. Change the reject count option.	1, START	*0 1*	**

Use the following procedure to select the reject count option from terminal mode:

Procedure	Terminal Key Sequence	Terminal Display

1. Select the reject count option.
(X is the number entered)
	5, X	`Command : 5 - Enter reject count option`
		`Programming reject count options -`
		`0 - Default`
		`1 - Optional`
		`Enter option:0`
		`Command :`

Select Verify Option

Three options are available for selecting verify and functional test routines. These options are:

Options	Description
0	Default option. Perform fuse verify, followed by structured test (if test vectors are present in RAM), and Logic Fingerprint test (if one or more Logic Fingerprint test cycles are selected), in that order.
1	Perform fuse verify only.
2	Perform structured test and Logic Fingerprint test only, in that order. Does not perform fuse verify.

Option 0 (default) is the option used in normal operation. Option 1 checks the programming of the device fuses without checking device functionality. Use option 2 to functionally test devices with the security fuse blown. In addition, option 2 can be used to learn the Logic Fingerprint test of a device with the security fuse blown. (Fuse data in RAM will be cleared during this operation.) Programming cannot occur with option 2 selected.

Verify options must be entered from either the programmer's keyboard or a terminal. The option will remain in effect until it is changed or until the unit is powered down. To reselect the default, key in option 0.

Use the following procedure select the functional test option from the front panel mode:

Procedure	Front Panel Key Sequence	Front Panel Display
1. Select the verify options.	SELECT, E, 0, START	*0 0*

As an example, from the options described above

2. Select functional test.	2, START	*0 2*	**

Use the following procedure to select the functional test option from the terminal.

Procedure	Terminal Key Sequence	Terminal Display
1. Select the verify options.	0, 2	See the following display.

```
Command : 0 - Enter verify option

0 - Sequence - fuse verify, structured test, Logic Fingerprint
1 - Fuse verify only
2 - Sequence - structured test, Logic Fingerprint

Enter verify option: 2
 Command :
```

Select Security Fuse Option

Some logic devices are equipped with protective fuses called security fuses. Once the security fuses are programmed, the fuse states in the logic array cannot be copied. Programming the security fuses makes it very difficult to pirate a device design.

NOTE
Refer to your P/T adapter User Note for adapter specific security fuse information.

With the LogicPak you can either enable programming of the security fuse at all times, allow programming only when security fuse data are downloaded to the PLDS via the serial port, or disable programming completely, whether security fuse data are downloaded or not.

When the security fuse has been blown, a Logic Fingerprint test and structured test can still be performed, but a fuse verify operation is not possible.

To enable programming of security fuses, two conditions must be met: 1) the security fuse state in the programmer RAM must be 1 (or true), and 2) security fuse programming must be enabled. Once the security fuse option is selected, it will remain in effect until changed or until the programmer is turned off.

When security fuse data are entered into RAM in the JEDEC ASCII-logic format, data in the G field indicate the state of the security fuse. The G field does not affect the enable state of the security fuse option. The enable state must be entered separately. This can be done before or after loading JEDEC ASCII-logic format data.

Security fuse states cannot be loaded from a master device.

CAUTION

Once the security fuse is blown, you cannot verify the state of any fuse in the device. For devices which are not erasable the process cannot be reversed; therefore, be certain that you want to program the security fuse before you activate this function. Attempting to reprogram the device after the security fuse is blown will alter the original fuse pattern and render the device inoperative.

Security Fuse Select-Code Options:

Option	Description
0	Default option. Disable programming and set the security fuse state in RAM to 0 (unprogrammed).
1	Disable programming and set security fuse state in RAM to 1 (programmed).
2	Enable programming and set security fuse state in RAM to 0. (Data downloaded in the JEDEC format can change the security fuse state to 1.)
3	Enable programming and set security fuse state in RAM to 1. (Data downloaded in the JEDEC format can change the security fuse state back to 0.)

Use the following procedure to select the security fuse option from front panel mode.

Procedure	Front Panel Key Sequence	Front Panel Display
1. Select security fuse option.	[SELECT] [E] [7] [START]	*0 0*

As an example, use the following procedure to enable security fuse programming and set security fuse state in RAM to 1 (option 3).

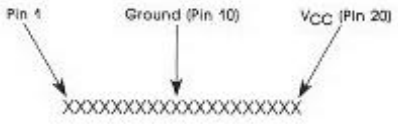
Procedure	Front Panel Key Sequence	Front Panel Display
1. Set security fuse state to 1.	[3] [START]	*0 3* **

Procedure	Terminal Key Sequence	Terminal Display
1. Select security fuse option.	[7] [3]	Command : 7 - Enter security fuse option OPTION SECURITY FUSE DATA SECURITY FUSE PROGRAMMING 0 -------- 0 -------- disabled 1 -------- 1 -------- disabled 2 -------- 0 -------- enabled 3 -------- 1 -------- enabled Enter Security Fuse option: 3 Command :

Enter Functional Test Data

Functional test data includes information for the Logic Fingerprint test and also the test vectors used by P/T adapters for testing of a programmed device. The Logic Fingerprint test information consists of three components:

- The number of test cycles to be performed during the Logic Fingerprint test (described in Appendix B). The default value is 00, which disables the Logic Fingerprint test.
- The Logic Fingerprint starting vector. This is an arbitrary binary sequence, each bit of which corresponds to a pin on the device under test. The starting vector format for a 20-pin device is shown below. Each "X" represents a "1" or a "0" to apply a logic high or logic low to the corresponding pin. The default value is all 0's. Values entered for V_{CC} and ground have no effect on the device under test.



- The starting vector is used to initialize the Logic Fingerprint, and is one of the components (along with the device type, number of test cycles, and fuse pattern) which determine the resulting Logic Fingerprint test signature. Note that different Logic Fingerprint test signatures may result for a given logic design, depending on the choice of starting vector.
- The Logic Fingerprint test signature itself is the result of performing the Logic Fingerprint test, as described later in this section.

Logic Fingerprint test data may be entered from either the front panel or the terminal. From the front panel, the number of test cycles and the starting vector for the Logic Fingerprint test may be entered, and the resulting Logic Fingerprint test signature may be viewed or entered. Structured test vectors may not be entered or edited from the front panel but only from a terminal or serial download. All functional test data may be entered from the terminal, including number of test cycles, starting vector, the Logic Fingerprint test signature itself, and the test vectors.

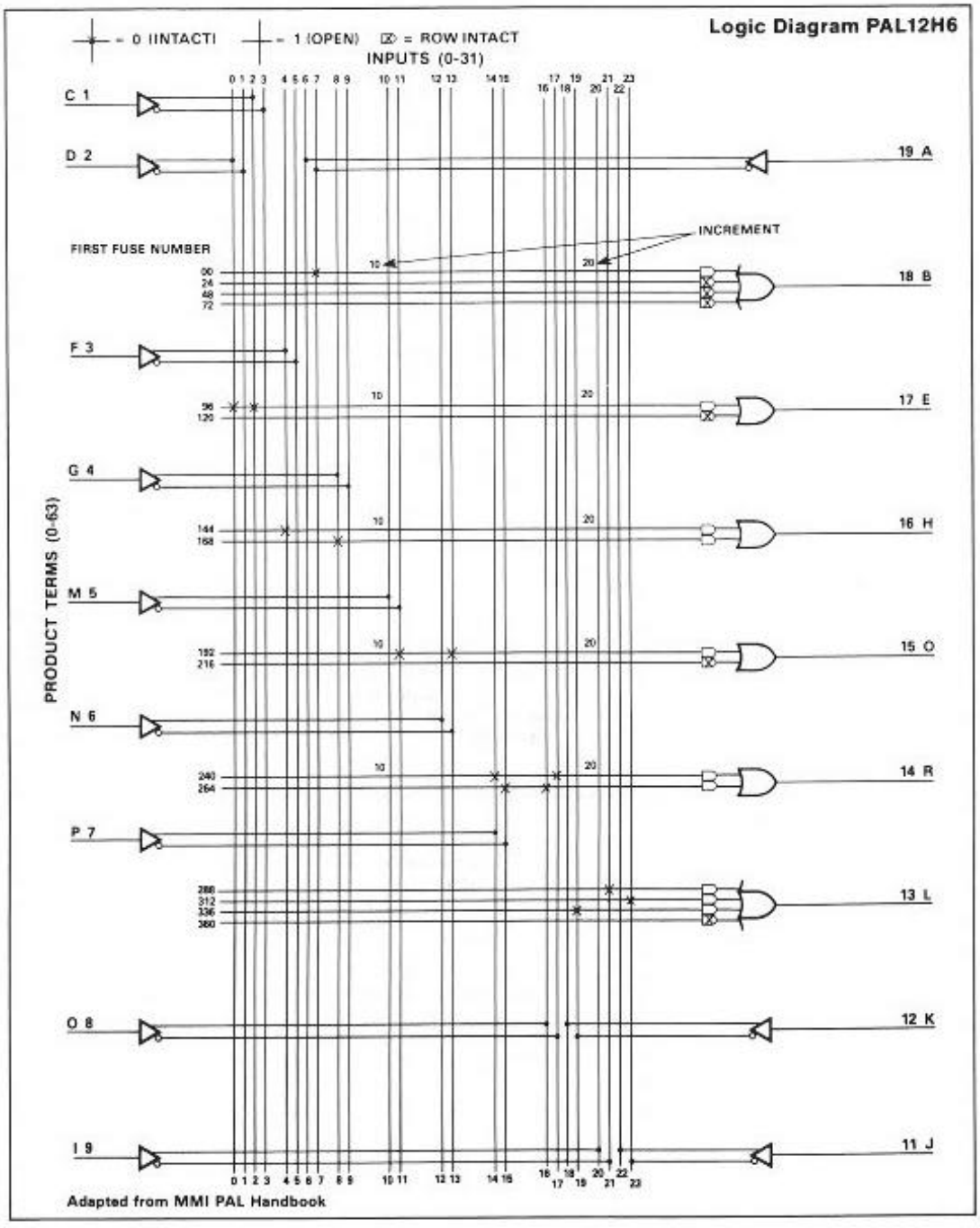
NOTE

If a value is entered for the Logic Fingerprint test signature, it should be either O O O O O O O O or a known-good value corresponding to the number of test cycles, starting vector, device, and fuse patterns under test. A value of O O O O O O O O will cause the LogicPak to learn the correct Logic Fingerprint test signature when a Load, Program, or Verify operation is performed. When in Load, the correct Logic Fingerprint will be learned independently of the value entered.

If "Device Selection Error" (Error 30) appears when you select functional test data, you must specify family code and pinout code to define the starting vector width.

In the subsections which follow, functional test data will be entered to test the Basic Gates design example[a]. This figure shows an example of one method of programming a PAL. Structured vectors, fuse map display, JEDEC file, and sum-check are functions that are illustrated in the Basic Gates example[a].

[a]Adapted from the MMI PAL HANDBOOK, available from Monolithic Memories, Inc., 1165 Arques Avenue, Sunnyvale, California 94086.


Logic Diagram PAL12H6
= 0 (INTACT)
= 1 (OPEN)
= ROW INTACT
INPUTS (0-31)
C 1
D 2
19 A
INCREMENT
FIRST FUSE NUMBER
00
24
48
72
18 B
F 3
96
120
17 E
G 4
144
168
16 H
M 5
192
216
15 O
N 6
240
264
14 R
P 7
288
312
336
360
13 L
O 8
12 K
I 9
11 J
PRODUCT TERMS (0-63)
Adapted from MMI PAL Handbook

From the front panel the number of test cycles and the Logic Fingerprint starting vector may be entered, and the Logic Fingerprint test signature may be viewed or entered. Use the following procedure to set the number of Logic Fingerprint Test cycles.

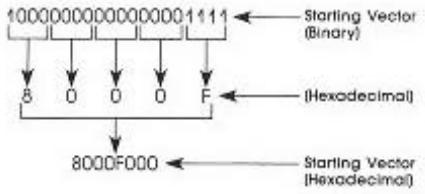
Procedure	Front Panel Key Sequence	Front Panel Display
1. Select the function to set the number of Logic Fingerprint Test cycles.	SELECT E 0 START	*0 0*

As an example, use the following procedure to enable one cycle of testing.

Procedure	Front Panel Key Sequence	Front Panel Display	
1. Enable one cycle of testing.	0 1 START	*0 1*	**

Logic Fingerprint and Starting Vector

The starting vector must be converted from the binary form to hexadecimal for entry from the front panel. For our Basic Gates example, we will choose an arbitrary test vector as shown:



Use the following procedure to select the Logic Fingerprint and starting vector function. The unused portion of the 32-bit vector is assumed to be zeroes and must be included in the hexadecimal vector entry.

Procedure	Front Panel Key Sequence	Front Panel Display
1. Select the starting vector function.	SELECT E 0 START	*0000 B1*
NOTE *The eight-character starting vector is entered into the programmer in two fields. B1 identifies the first field.*		
2. Enter the first four hexadecimal digits.	8 0 0 0	*8000 B1*
3. Select the B2 field.	START	*0000 B2*
NOTE *B2 represents the second field.*		
4. Enter the remaining hexadecimal digit.	F 0 0 0	*F000 B2*
5. Display the starting vector.	START	
NOTE *The zeros are ignored, but are needed to correctly position the "F". This vector, when applied to the Basic Gates example, produces the Logic Fingerprint test signature: ED37A9E4 (hexadecimal).*		
6. Enter the fingerprint test.	E D 3 7	*ED37 E1*
NOTE *The first four characters are displayed as the E1 field.*		
7. View the E2 field.	START	
8. Enter the values for the E2 field.	A 9 E 4	*A9E4 E2*
9. Display the E2 field.	START	*A9E4 E2* **

Use the following procedure to select the Logic Fingerprint and starting vector from the terminal.

Procedure	Terminal Key Sequence	Terminal Display
1. Select the function to enter functional test data.	8	See the following display.

```
Command : 8 - Enter functional test data

Cycles for Fingerprint: 01
Fingerprint starting vector: 1000000000000001111
Fingerprint: ED37A9E4
```

As each prompt appears, you may modify the current values using the following steps:

1. Move the cursor forward (using the spacebar) and backward (using the backspace) along the displayed value until it is positioned over the symbol to be changed.
2. Press the desired symbol.
3. Enter RETURN at any point to move to the next prompt.
4. CTRL Z is used to exit the functional test data entry mode.

Vector Editing

Vectors are created in RAM by downloading JEDEC "V" fields, simulating a source file containing a function table (using PALASM), or by using the vector editor in the terminal mode. (Function 8 on the menu.)

When the Logic Fingerprint test information has been entered (or skipped by entering RETURN), the vector editor menu appears (see following example), and a prompt appears for the vector number to be edited. The default vector is 0001, as shown in the next example.

NOTE

Some P/T adapters support both the dual-in-line (DIP) package and the chip carrier package. Generally, the chip carrier has more pins than the DIP socket. Structured vectors must be written for the DIP socket and the pinout is automatically carried across to the functionally identical pinout on the chip carrier.

```
Command : 8 - Enter functional test data

Cycles for Fingerprint: 01
Fingerprint starting vector: 1000000000000001111
Fingerprint: ED37A9E4

          - DISPLAY -                    - EDITING COMMANDS -
0 ------------ Display menu          D ------- Delete (Kill) current vector
Return -------- Go to next vector     R ------- Repeat current vector
U ------------- Up (previous vector)  CTRL Z -- Exit vector editor
#(N) ---------- Go to vector (N)
Space --------- Move cursor right
BKSP (CTRL H) - Move cursor left
CTRL Z -------- Exit vector editor

Edit structured vector:  0000
0001:  1100XX10XNXXXHXLHH8N
0002:  --------------------
```

The vector editor is a fixed-format line editor with the first column of the displayed line reserved for command characters, as shown in the next example.

A character entered in the first column (normally blank) is interpreted as a command and acted upon immediately; otherwise, vector editing is not processed until a RETURN is entered (at any point on the line). The command characters recognized in the first column are 0, U, #, D, and R; see the table for command character definitions.

Command	Description	Activity
0 (zero)	Display menu	Redisplays menu and restarts editing on the same vector.
U	Up (previous vector)	Moves editing to the next lower vector number (the vector one 'up' on the screen).
# (N)	Go to vector (N)	Entering a # in the command column causes the vector editor to prompt for the desired vector number (default = 0001). Entering a vector number greater than the last vector will move you to the last vector.
D	Delete current vector	Current vector is deleted, and all higher vectors moved down one. Current vector number is redisplayed with new vector.
R	Repeat current vector	Creates a copy of the current vector immediately following the current vector. The copy is displayed, with its vector number (one greater than the original). This command may be given for any vector, and existing vectors will be moved to accommodate the new copy.

During operation, the vector editor copies the selected vector to a temporary buffer where all editing changes are made. Then, when a RETURN command is entered, the temporary buffer is examined for legal characters before copying back to vector memory. You are not allowed to proceed to another vector until all characters are legal in the current vector. Typing a CTRL Z to exit the vector editor will leave the selected vector in its original state. An *empty vector* is represented by a dash in all pin positions. This will appear as the first vector in an empty vector editor buffer, or as one past the last vector where data are present in memory. All vectors are numbered lower than the empty vector.

Use the following procedure to edit a vector.

Procedure	Keystrokes	Definition
1. Increment (spacebar) or decrement (backspace) by one position.	spacebar backspace	Moves the cursor forward or backward along the test vector.
2. Select the desired test conditions to enter into the vector image from the allowable characters.	0	Drive input low.
	1	Drive input high.
	2-9	Drive input to super-voltage #2-9.
	C	Drive input low, high, low.
	K	Drive input high, low, high.
	N	Power pins and outputs not tested.
	L	Test output low.
	H	Test output high.
	Z	Test output for high impedence.
	F	Float input or output.
	*X	Ignore input or output (not JEDEC format).
	P	Preload (applied to clock pin).

*X is not defined in the JEDEC format. The X is treated as an N for outputs and leaves an input at its previously defined state.

NOTE

Test conditions 2 through 9 specify non-TTL levels (supervoltages) that access special device features. A device may be damaged by improper use of supervoltages.

3. Move the cursor to the next vector or exit the editor.	RETURN	At any point moves the cursor to the next vector.
	CTRL Z	Exits the vector editor.

Register Preload

In some registered logic devices, the internal registers can be arbitrarily loaded to a desired state. This capability allows easier functional testing by providing a means of achieving states which may be difficult or impossible to enter by normal state transitions.

For devices which have the register preload feature preload is accomplished by using a "preload vector", a structured vector which has a "P" symbol in the clock pin position. Also in the preload vector are special symbols in the positions of the pins associated with loading of the registers. The symbols used in the preload vector and their functions are described in the following table.

Preload Vector Symbols

P	Identifies preload vector and invokes preload algorithm. (Allowed on clock pin only, otherwise treated as "X".)
O	Preloads a logic "0" into the register $\overline{Q}$ output, meaning a logic "1" wil be loaded into the register Q output. Does not test device outputs.
1	Preloads a logic 1 into the register $\overline{Q}$ output, meaning a logic O will be loaded into the register Q output. Does not test device outputs.
L	Preloads register with the appropriate level such that a logic O appears on the device output pin. Also tests the preloaded device output and indicates an error if a logic O is not found. Not allowed for some devices.
H	Preloads register with the appropriate level such that a logic 1 appears on the device output pin. Also tests the preloaded device output and indicates an error if a logic 1 is not found. Not allowed for some devices.

For adapter specific information on devices which have the preload feature, refer to your adapter User Note.

All pins not used in the device's preload algorithm (regardless of the symbol placed in the preload vector pin position) are treated as "X"s (left in their previous state). Pins which are used in the preload algorithm may not return to their original state following preload. For example, to preload a 20-pin device with preload pins (most likely device outputs) 12 through 19, you might apply the following preload vector: (clock pin assumed to be pin 1).

0001: PXXXXXXXXNXHLHLHLHLN

When the preload vector is applied during functional testing, the device-specific preload algorithm is invoked and the registers are loaded with the appropriate data to make the outputs high "H" or low "L". The output pins are then tested to verify that the preload was successful.

Assuming the device has an inverter between the register output and the output pin, another method of achieving the same results as above is to use the following two vectors:

0001: PXXXXXXXXNX10101010N

0002: XXXXXXXXXN0HLHLHLHLN

The first vector is a preload vector using "1"s and "0"s to load the Q output of the register with the data indicated (thus making the $\overline{Q}$ outputs of the registers the complements of the data in the vector). Since we have assumed an inverter between the $\overline{Q}$ output and the output pin, the data found on the output pins after execution of the preload vector should reflect the "1"s and "0"s in the preload vector. The second vector shown is a conventional structured vector which tests the outputs for the desired data.

The "1" and "0" preload symbols are most useful for preloading registers whose state cannot be read at a device pin, or for any case in which the user is concerned with setting up the state of the registers not necessarily the state of the output pins.

The H&L preload symbols are used to preload the states of output pins whose states are determined by the data in internal registers. The P/T adapter firmware determines what data should be placed in the internal registers to provide the correct outputs. Users concerned with preloading the state of the internal registers can use the H&L preload vector to load and automatically verify internal register states provided that data inversion (if any) between registers and outputs is considered. Some devices depend upon a fuse state to determine the data inversion (if any) between the registers and the outputs. If this is the case, the H&L style preload vector is not allowed and an error will result.

Display and Transmit Fuse Pattern

This command transmits the fuse pattern in the programmer data RAM to the serial port. The fuse states may be shown as a series of "1"s and "0"s or a series of "–"s and "X"s; see the subsection on selecting characters. The "1" or "–" represents a high-resistance or "blown" fuse in a fuse link device. The "0" or "X" represents a low-resistance or "intact" fuse. Each fuse can be identified by a decimal fuse number, as shown in the figure. The fuse states are arranged in a matrix that corresponds to the logic diagram of the device (see the figure for the Logic Diagram for Basic Gates Example). This is useful for comparing or copying a displayed fuse pattern to the device logic diagram.

```
Command : R - Display fuse pattern

      00          10
0000  XXXXX---XX  -XX--XX-XX
0020  XXXXX-----  -----XXXXX
0040  ---------X  XXXXXXX--
0060  XXX----XXX  X----XXXXX
0080  ----------  -----XXXXX
0100  XXXX------  ----------
0120  --------X-  ---------X
0140  XXXXXXXXXX  XXXXXX---
0160  XXXXXXXXXX  XXXXXXXXX
0180  ------XXX-  --------XX
0200  XXXXXXXXXX  XX-X-X-X-X
0220  XXXXXXXXXX  X-----XXXX
0240  -------X-X  XXX--XXXXX
0260  -----XXX--  --XXXX--XX
0280  XXXXXXXXXX  -XXXXXXXX
0300  XXXXXXXXXX  XXXXXXXXX
Surcheck 10B0

Command :

Note: - = open
      X = intact

Fuse Number = First Fuse Number + Increment
```

NOTE

Sending certain control characters to the PLDS during the course of fuse pattern display will affect the display. The output may be stopped by sending a CONTROL S (DC1 or ASCII, 11 hex) and then restarted by sending a CONTROL Q (DC3 or ASCII 13 hex).

A CONTROL Y (ASCII, 19 hex) will terminate the transmission and return to the terminal or front panel operation.

An ESC (escape) character (ASCII 1B hex) will also terminate the transmission and return to front panel operation.

If the underblow/overblow attribute is enabled, the fuse map display may contain some U's (underblow) or B's (overblow). See the subsection on Select Attributes for the definition of underblow and overblow.

The last character of the fuse pattern transmission is either CONTROL C (ETX or ASCII 03) or a CONTROL Z (ASCII 1A hex). (See the section on Select Attributes.)

Use the following procedure to display the fuse pattern on the terminal screen from front panel mode.

Procedure	Front Panel Key Sequence	Front Panel Display
1. Display the fuse pattern.	SELECT E A START	XXXX ** ⍰

NOTE
⍰ is the action symbol. XXXX is the fuse data checksum.

Use the following procedure to display the fuse pattern on the terminal screen from the terminal.

Procedure	Terminal Key Sequence	Terminal Display
1. Display the fuse pattern	A	Command : A - Display fuse pattern

```
      00          10
0000  XXXXX---XX  -XX--XX-XX
0020  XXXXX-----  -----XXXX
0040  ---------X  XXXXXXX--
0060  XXX----XXX  X----XXXX
0080  ----------  -----XXXX
0100  XXXX------  ----------
0120  --------X-  ---------X
0140  XXXXXXXXX  XXXXXX---
0160  XXXXXXXXX  XXXXXXXXX
0180  ------XXX-  --------XX
0200  XXXXXXXXX  XX-X-X-X-X
0220  XXXXXXXXX  X-----XXXX
0240  -------X-X  XXX--XXXX
0260  -----XXX--  --XXX--XX
0280  XXXXXXXXX  -XXXXXXXX
0300  XXXXXXXXX  XXXXXXXXX
Suncheck 1080
```

JEDEC Format Data Exchange

Transmit JEDEC Data

This command transmits the contents of the fuse and vector RAM to the serial port in the JEDEC format (see Appendix C).

The following characteristics apply to JEDEC transmission:

- ☐ The output may be halted by sending a CONTROL S (DC1 ASCII, 11 hex) and restarted by sending a CONTROL Q (DC3 or ASCII, 13 hex).
- ☐ An ESC character (ASCII, 1B hex) will abort the transmission and return to the programmer front panel operation.
- ☐ A CONTROL Y (ASCII, 19 hex) will terminate the transmission and return to the terminal or programmer front panel operation.
- ☐ The Logic Fingerprint test fields (S, R, and T) are not sent if the number of cycles is 0.
- ☐ The G field is sent only if security fuse data is a 1.
- ☐ The fuse checksum (C field) is the 16-bit sum of all fuse states (i.e., from fuse 0 to the fuse limit for the device). See the following example.

```
(STX)*F0*L0000
01001110 00001000 11110000 11111111 01010001*
C021A*
(ETX)0000

The F0* cleared all the fuse RAM to 0. The L field transmitted 40
fuse states starting at 0.

Fuse number 00 01 02 03 04 05 06 07  08 09 10 11 12 13 14 15  16
State        0  1  0  0  1  1  1  0   0  0  0  0  1  0  0  0   1

                MSB          LSB

                 7 6 5 4 3 2 1 0

0000             0 1 1 1 0 0 1 0          72
0008             0 0 0 1 0 0 0 0          10
0016             0 0 0 0 1 1 1 1          0F
0024             1 1 1 1 1 1 1 1          FF
0032             1 0 0 0 1 0 1 0          8A
0040             0 0 0 0 0 0 0 0          00
0048             0 0 0 0 0 0 0 0          00
XXXX             0 0 0 0 0 0 0 0          00
                                          ----
                                          021A
```

Use the following procedure to transmit JEDEC data from front panel mode.

Procedure	Front Panel Key Sequence	Front Panel Display
1. Select the transmit data function.	SELECT E C START	XXXX ** ☐

NOTE
☐ is the action symbol. XXXX is the fuse data sumcheck.

Use the following procedure to transmit JEDEC data from the terminal.

Procedure	Terminal Key Sequence	Terminal Display
1. Select the transmit data function.	C	See the following display.

```
 Command : C - Transmit JEDEC data
PAL12H6          PAL DESIGN SPECIFICATION
P7000             JANE ENGINEER   5-21-83
BASIC GATES
DATA I/O
*
QP20*
QF0384*
L0000
111111101111111111111111
00000000000000000000000000
00000000000000000000000000
00000000000000000000000000
01011111111111111111111111
00000000000000000000000000
1111011111111111111111111
1111111101111111111111111
1111111111101011111111111
00000000000000000000000000*
L0240
1111111111111101101111111
1111111111111110011111111
1111111111111111111111011
1111111111111111111111110
1111111111111111111101111
00000000000000000000000000*
V0001 XXXXXXXXXNXXXXXXXHXN*
V0002 XXXXXXXXXNXXXXXXXL1N*
V0003 X0XXXXXXXNXXXXXXL1XN*
V0004 X1XXXXXXXNXXXXXXL1XN*
V0005 1XXXXXXXXNXXXXXXL1XN*
V0006 11XXXXXXXNXXXXXXHXXN*
V0007 XX00XXXXXNXXXXXXLXXXN*
V0008 XX01XXXXXNXXXXXXHXXXN*
V0009 XX10XXXXXNXXXXXXHXXXN*
V0010 XX11XXXXXNXXXXXXHXXXN*
V0011 XXXXXXXXXN00NXXXXXXXN*
V0012 XXXXXXXXXN01HXXXXXXN*
V0013 XXXXXXXXXN10HXXXXXXN*
V0014 XXXXXXXXXN11HXXXXXXN*
V0015 XXXXXXXXXNSJLXXXXXXN*
V0016 XXXX00XXXNXXXHXXXXN*
V0017 XXXX01XXXNXXXLXXXXN*
V0018 XXXX10XXXNXXXLXXXXN*
V0019 XXXX11XXXNXXXLXXXXN*
V0020 XXXXXX00XNXXXLXXXXN*
V0021 XXXXXX01XNXXXHXXXXN*
V0022 XXXXXX10XNXXXHXXXXN*
V0023 XXXXXX11XNXXXLXXXXN*
T01*
S0000000000000000000*
RRB472BBB*

C1BB9*
2642
```

Receive JEDEC Data

This command prepares the programmer to receive JEDEC formatted fuse and vector data from a peripheral device via the serial port.

NOTE

The D field is ignored by the translator. The correct family and pinout code must be entered before receiving JEDEC data.

Three types of errors may be caused by receiving improper data in the JEDEC format (see the next table).

Translator Input Error Codes

Error	Description	Possible Fields
82	SUMCHK ERR	Transmission checksum
84	INVALID DATA	EXT, F, L, S, V
91	I/O FORM ERR	C, G, L, P, R, T, V

You may determine the field in which the error occurred by examining data RAM location 0408; the ASCII value (hexadecimal) of the field is stored here (see the following table). More information about the possible cause of the error may be found in the table on Translator Input Error Codes.

Field Identifier	ASCII Character Hex Value
(EXT)	03
C	43
F	46
G	47
L	4C
P	50
QF/QP	51
R	52
S	53
T	54
V	56

Use the following procedure to examine the data RAM location 0408 from front panel mode.

Procedure	Front Panel Key Sequence	Front Panel Display
1. View the RAM address.	EDIT	*EDIT ADDR XXXX*
	4 0 8 START	*0408 D**^RXX*

NOTE

XXXX is the current address. XX is the field identifier in hexadecimal.