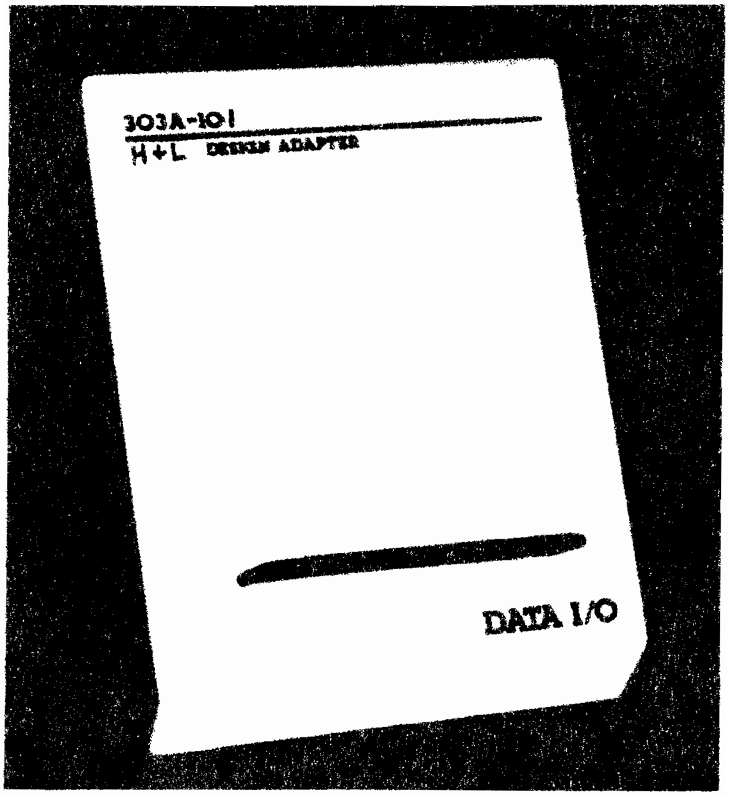


303A-101 H + L DESIGN ADAPTER 715-1951

(303A-100)

303A-101 H&L
DESIGN ADAPTER MANUAL

10-715-1951 Version 02 Rev. Z Aug83
Applies to configuration 715-1951-003

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 any damages that result from use of this document or the equipment which it accompanies.

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

TABLE OF CONTENTS

SECTION 1. INTRODUCTION

SECTION 2. INSTALLATION AND MAINTENANCE

SECTION 3. DATA DEVELOPMENT

LIST OF FIGURES

LIST OF TABLES

SECTION 1. INTRODUCTION

1.1 GENERAL INFORMATION

The 303A-101 H&L design adapter consists of interface circuitry and EPROM memory, mounted in a metal frame. It is a software development aide used with the Data I/O LogicPak to enable you to define logic structure and reduce it to a fuse pattern. The 303A-101 H&L Design Adapter is a subsystem of the Programmable Logic Development System (PLDS). The PLDS was created to provide a means to develop data, program, verify, and test currently available logic devices. For more information see the PLDS Overview Manual (Data I/O part number 10-950-1942-001).

The information in this manual is divided into three parts. The introduction section gives you general information on the design adapter and specifics on field applications support, warranty, service, and ordering. The installation section tells how to install the adapter. The data development section describes the operations used to develop state tables and transform them into a fuse pattern. Read the manual thoroughly before attempting to develop data with the 303A-101 H&L design adapter.

1.2 APPLICATIONS

The design adapter helps you define the logic structure that you intend to program. It is usually used in conjunction with a terminal connected to the programmer. Its firmware translates your inputs, including part numbers and logic descriptors, into a fuse pattern and testing format. The resultant logic structure is contained in the programmer RAM. This enables the design adapter to be removed from the LogicPak, the correct programming/testing (p/t) adapter to be installed, and the fuse pattern programmed directly into the device.

Alternately, the logic structure can be offloaded at any development stage for extended storage via (1) the serial port to disk or tape or (2) a p/t adapter to a 4K MOS PROM.

When the design adapter is installed on the LogicPak, normal Load, Program, and Verify operations are disabled.

The Data I/O Programmable Logic Applications Chart in the back of this manual lists all the logic devices currently available and those compatible with the 303A-101 Design Adapter, as well as the necessary PLDS components used to develop, program, verify, and test each. When new devices become available, the chart will be updated.

1.3 FIELD APPLICATIONS SUPPORT

Data I/O Field Application Engineers (FAE's) throughout the world can provide additional information about interfacing Data I/O products with other equipment, and answer questions about your equipment.

The locations of the FAE's within the United States are given on the address list at the back of this manual. For international applications support, contact your nearest Data I/O representative.

1.4 WARRANTY

Data I/O equipment is warranted against defects in materials and workmanship. The warranty period of ninety days begins when you receive the equipment.

The warranty card inside the back cover of this manual explains the length and conditions of the warranty. For warranty service, contact your nearest Data I/O Service Center.

1.5 SERVICE

Data I/O maintains service centers throughout the world staffed with factory-trained technicians to provide prompt, quality service. A list of all service centers is located at the back of this manual.

1.6 ORDERING

To place an order for equipment, contact your Data I/O sales representative. Orders for shipment must include the following information:

- o Description of the equipment. (See the latest Data I/O Price List or contact your sales representative for equipment and part numbers.)
- o Purchase order number.
- o Desired method of shipment.
- o Quantity of each item ordered.
- o Shipping and billing address of the firm, including zip code.
- o Name of person ordering equipment.

SECTION 2. INSTALLATION AND MAINTENANCE

2.1 INSPECTION

The 303A-101 H&L Design Adapter (fig. 2-1) was thoroughly tested and inspected before shipment. For trouble-free operation, verify on receipt that it is in the best possible condition. It was carefully packaged before shipment and should arrive in good operating condition. All the equipment listed in table 2-1 should be present. Carefully inspect the adapter for any damage that may have occurred during transit. If you note any damage, file a claim with the carrier and notify Data I/O. Read the information in sections 2.2 and 2.3 before using the design adapter for data development.

2-1. Required Equipment

DESCRIPTION	DATA I/O PART NUMBER	QTY
303A-101 H&L Deisgn Adapter	715-1951-002	1
Instruction Manual	10-715-1951-002	1

2.2 INSTALLATION

Install the design adapter in the LogicPak as shown in figure 2-2.

CAUTION
WHEN CHANGING ADAPTER

BEFORE REMOVING THE ADAPTER, enter an <ESC> from the terminal; or from the programmer front panel press the KEYBOARD key (on the System 19) or the VERIFY key (on the 100A or 29A). The Processor in the programmer executes firmware resident in the adapter. These precautions must be taken before removing the adapter from the LogicPak to prevent a program interrupt and resulting loss of RAM data.

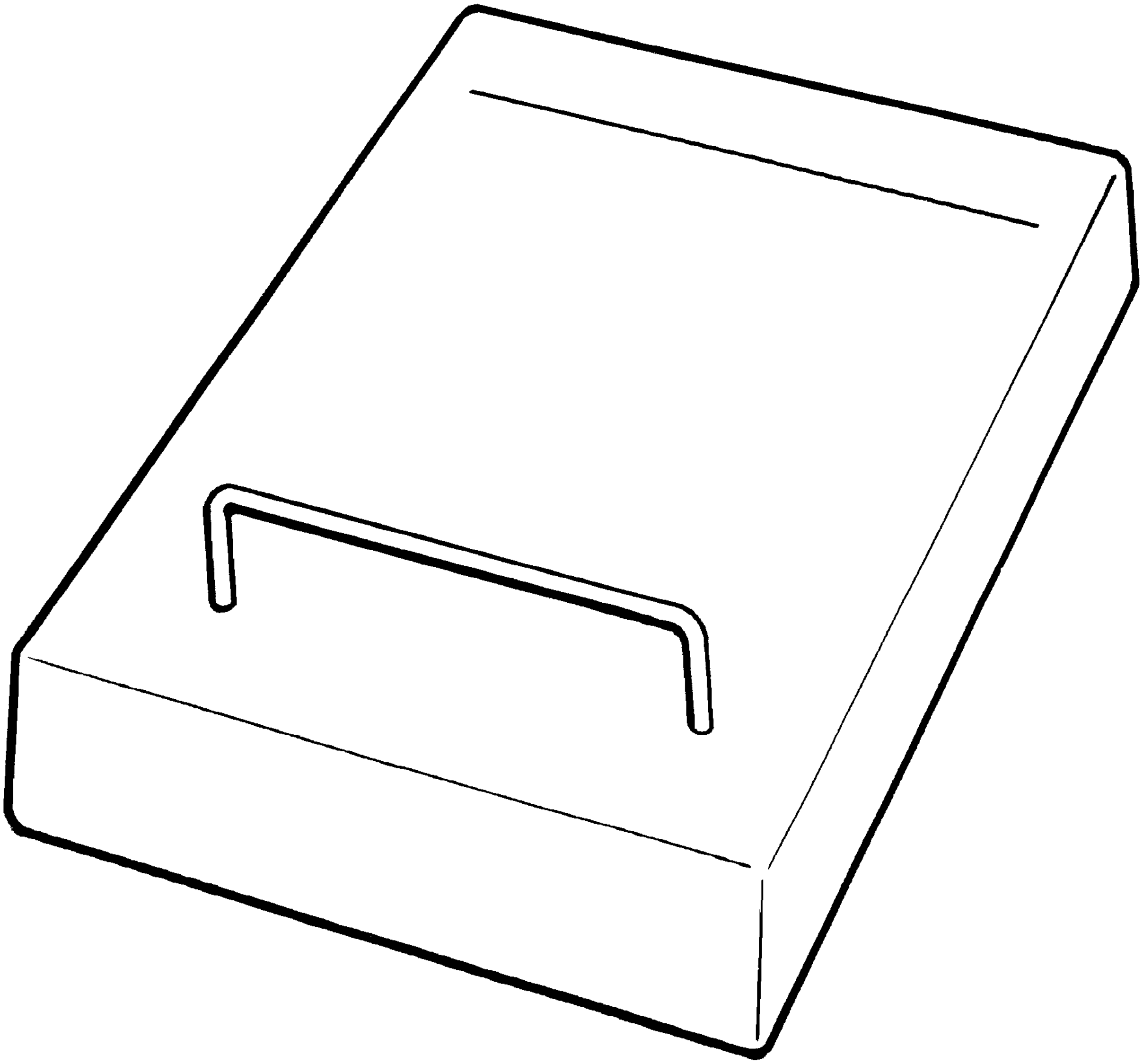


Figure 2-1. 303A-101 H&L Design Adapter

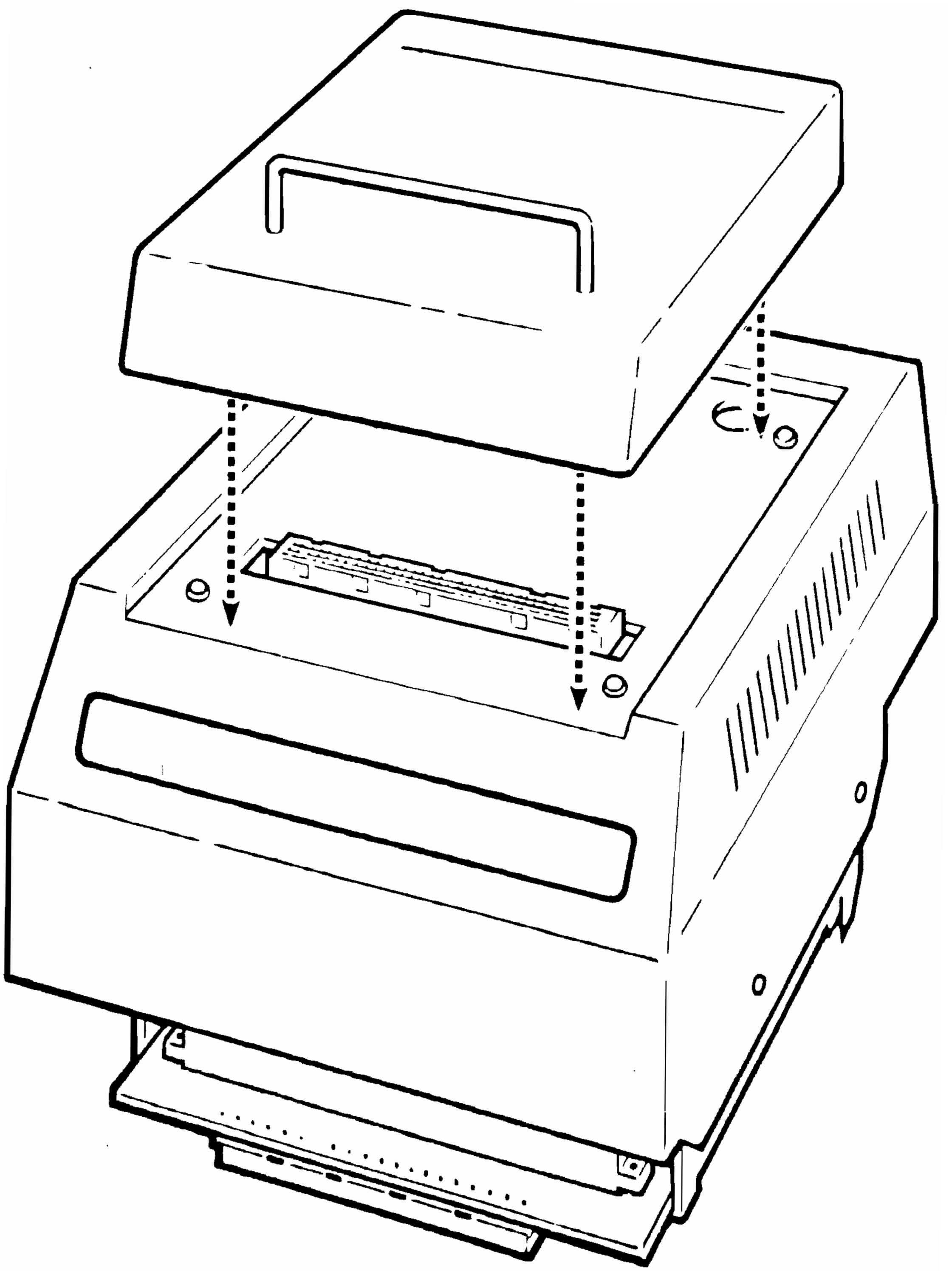


Figure 2-2. H&L Design Adapter Installation

SECTION 3. DATA DEVELOPMENT

3.1 INTRODUCTION

This section explains the steps necessary to develop data using the 303A-101 H&L Design Adapter. An overview of the H&L data development process is given in section 3.2. Section 3.3 gives the necessary preoperational procedures and section 3.4 discusses the function menu. Section 3.5 describes the step-by-step data development process using a terminal.

We recommend that you familiarize yourself with the commands, functions, and overall flow of events before starting to develop data with the 303A-101 H&L Design Adapter.

3.2 OVERVIEW

Figure 3-1 is a flowchart of the generalized data development process. Data development with the 303A-101 Design Adapter is best accomplished with a PLDS configuration consisting of the LogicPak, design adapter, terminal, and a Data I/O programmer. The terminal, though not essential for many operations, is required to enter design information and all other operations in the design adapter's repertoire that are not performed by the p/t adapter.

Programmable logic devices are arrays of logic elements joined by matrices of fusible links. You can custom design a logic circuit by blowing selected fuses in the matrices. Figure 3-2 shows the programming options of each fuse matrix and which fuses must be blown to create each logic function.

The Process. There are three basic steps to follow when you program a logic device:

1. Prepare a function (truth) table just as you would when designing with discrete logic.

2. Transfer the function table into the programmer.

3. Program a blank device.

NOTE

If you already have a programmed master device, you can load the data into the programmer and skip steps 1 and 2 given above. See your programmer manual for procedures.

The first step, preparing a function table, is common to any logic design process. Your function table cannot exceed the size of the device. The table should be formatted as sets of input variables and corresponding sets of output states. The details vary according to the device used. Each set of input and output conditions is considered a term or gate for data entry to the programmer.

The second step, transferring the function table into the programmer's data RAM, is covered in detail in this manual.

The design adapter simplifies the tedious step of translating logic structure into a fuse pattern that needs to be programmed in order for the device to perform its desired function.

Design adapter firmware allows you to specify a device, enter function tables (1) manually from the terminal keyboard or (2) from other sources via the RS-232 serial port in the programmer.

Firmware then translates the information into a fuse pattern in the programmer RAM. Editing can be performed at any stage at equation or fuse level. The development adapter also contains firmware for extensive test, data entry and verify options.

After data has been entered, the design adapter can be removed from the LogicPak and replaced with a (p/t) adapter. The device is then programmed with the fuse states existing in the programmer RAM.

The third step, programming a blank device, is covered in the manual for the specific programmer you are using.

NOTE

The memory map of the bit pattern in the programmer RAM versus the device fuse pattern was changed, beginning with the IFL LogicPak 950-0104-100.

The new organization is more universal and will accomodate many additional logic devices. If you are loading data into the programmer via the serial port using a PROM-type format (hex or binary, for example) you must prepare new paper tapes or data files in the ASCII-logic format or JEDEC format for use with the 303A LogicPak using the 303A-101 design development adapter or the

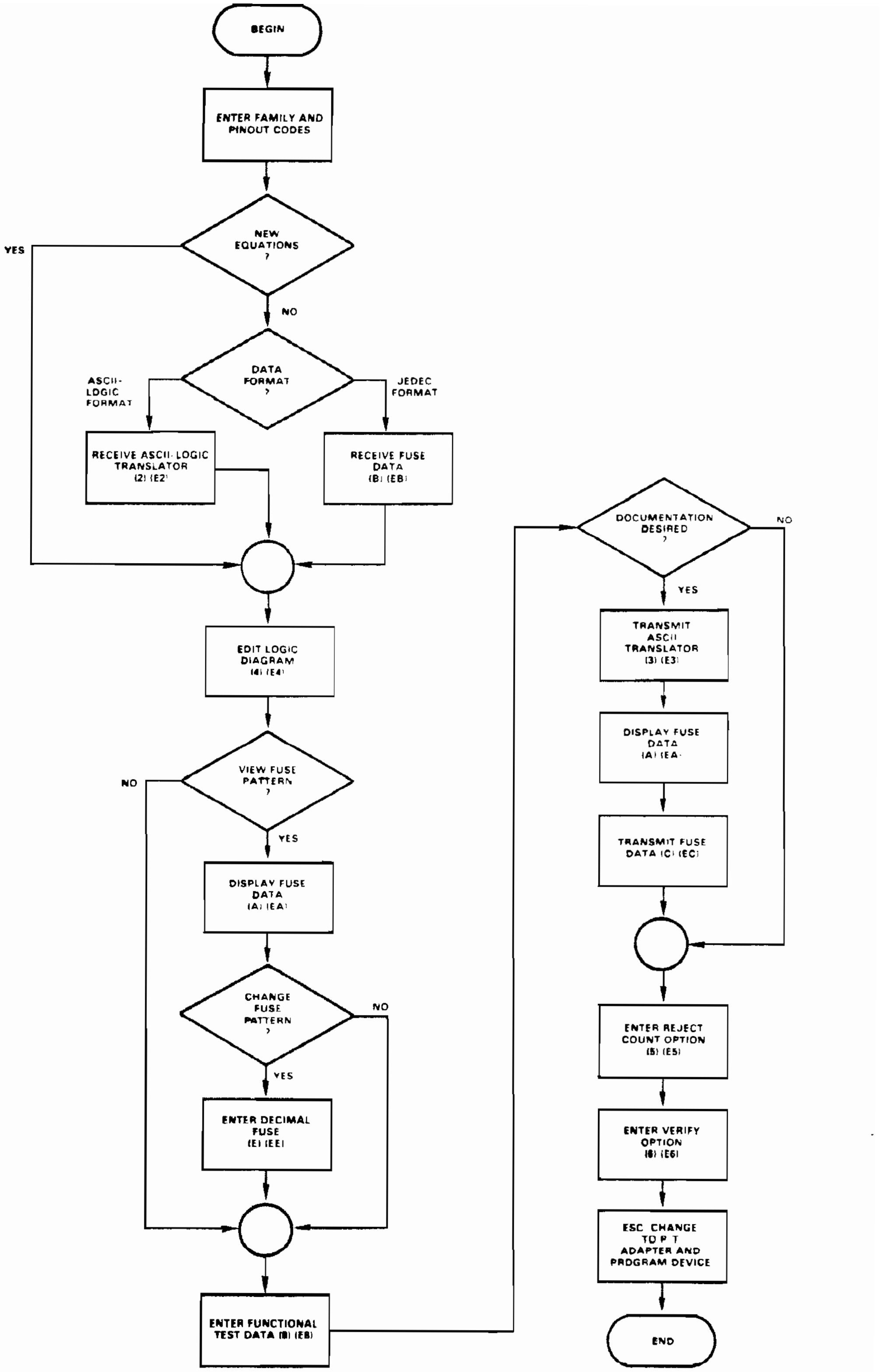
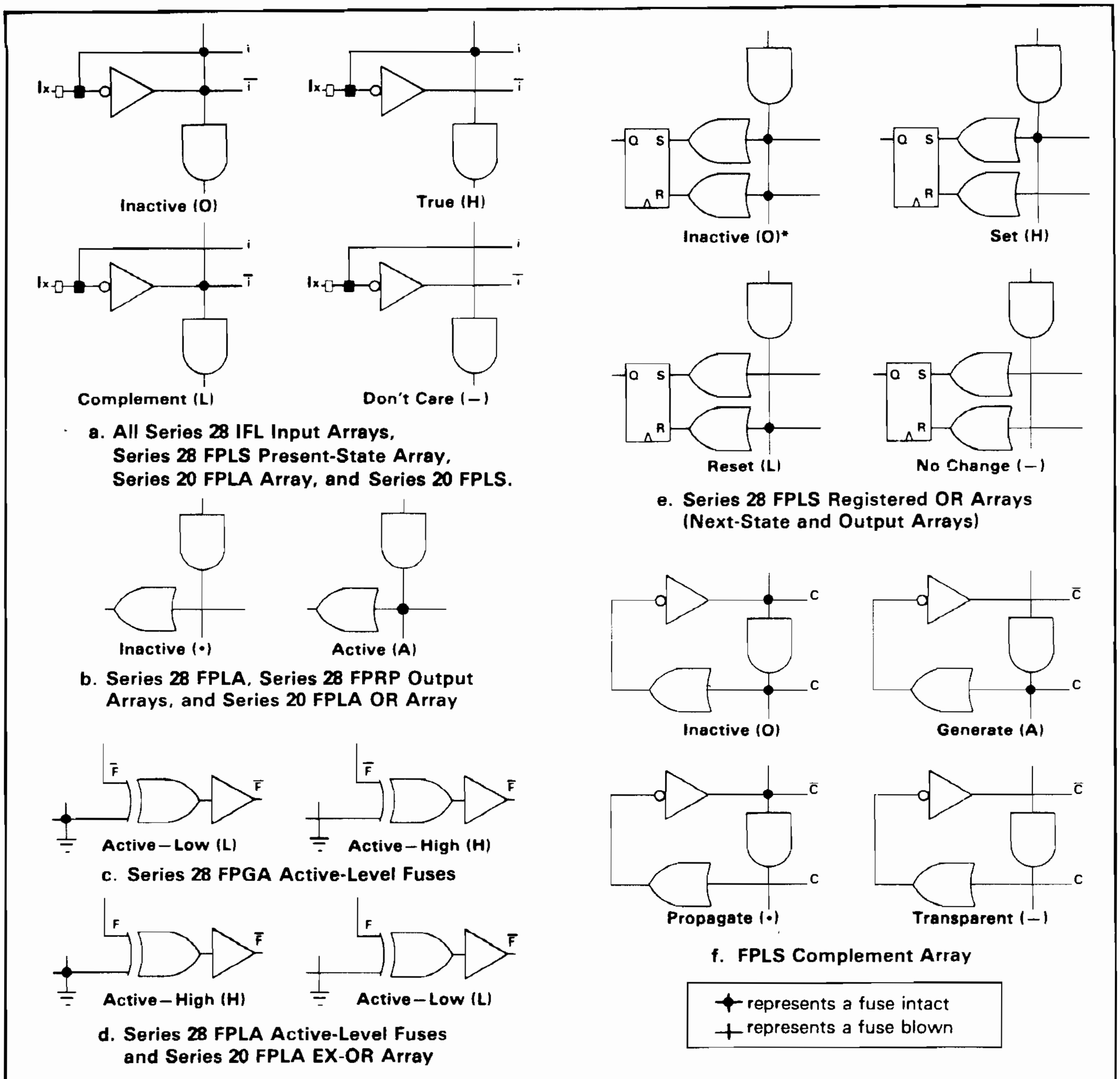

BEGIN
ENTER FAMILY AND PINOUT CODES
NEW EQUATIONS ?
YES
NO
DATA FORMAT ?
ASCII-LOGIC FORMAT
JEDEC FORMAT
RECEIVE ASCII-LOGIC TRANSLATOR (2) (E2)
RECEIVE FUSE DATA (B) (EB)
EDIT LOGIC DIAGRAM (4) (E4)
VIEW FUSE PATTERN ?
NO
YES
DISPLAY FUSE DATA (A) (EA)
CHANGE FUSE PATTERN ?
NO
YES
ENTER DECIMAL FUSE (E) (EE)
ENTER FUNCTIONAL TEST DATA (8) (E8)
DOCUMENTATION DESIRED ?
NO
YES
TRANSMIT ASCII TRANSLATOR (3) (E3)
DISPLAY FUSE DATA (A) (EA
TRANSMIT FUSE DATA (C) (EC)
ENTER REJECT COUNT OPTION (5) (E5)
ENTER VERIFY OPTION (6) (E6)
ESC CHANGE TO P T ADAPTER AND PROGRAM DEVICE
END

Figure 3-1. Flowchart, Data Development with 303A-101 H&L Design Adapter



a. All Series 28 IFL Input Arrays, Series 28 FPLS Present-State Array, Series 20 FPLA Array, and Series 20 FPLS.

b. Series 28 FPLA, Series 28 FPRP Output Arrays, and Series 20 FPLA OR Array

c. Series 28 FPGA Active-Level Fuses

d. Series 28 FPLA Active-Level Fuses and Series 20 FPLA EX-OR Array

e. Series 28 FPLS Registered OR Arrays (Next-State and Output Arrays)

f. FPLS Complement Array

* To prevent exciting both S and R flip-flop inputs simultaneously, this state is only allowed for OR gates driven by inactive AND gates.

Figure 3-2. IFL Programming Options

303-001 p/t adapter. Failure to do so will result in improperly programmed devices. If you are using master devices as your data source this change is transparent to you.

If you are loading data via the serial port in the ASCII-logic format used by the Model 10, the 950-0800, and the 950-0104, it will be accepted by the 303A LogicPak in the new memory map arrangement if a 303A-101 Design Adapter is installed.

New tapes or files can be prepared by first programming a master device using the earlier hardware and tapes. Next, install the 303A LogicPak and 303A-101 design adapter and load the master device into RAM. Then download the data out the port to your tape punch or other storage medium.

3.3 PREOPERATIONAL PROCEDURES

The following steps must be performed to ready the PLDS for data development with the 303A-101 Design Adapter.

1. Install the design adapter as described in Section 2 of this manual.
2. Set the parity, stop bits, and baud rate so they are the same on all the equipment in use.
3. Set terminal for:
   a) Disable any special functions using CNTL H, CNTL M, CNTL P or CNTL Z.
   b) Select distructive cursor for proper display during edit mode.
4. Power-up the programmer and terminal.
5. Press SELECT.
6. Enter EI.
7. Press START.
8. Enter the Family and Pinout Codes from the terminal. See Programmable Logic Applications Chart).
9. The function menu will be displayed.

The menu explanations in section 3.5 are intended to supplement the more complete discussion of these functions and commands in the LogicPak manual. Refer to the LogicPak manual for additional information when necessary.

Most commands can be entered with a single keystroke from the terminal, and the sequence requires no explanation. Options that require entry of additional data are described in text as they would be performed from a terminal. Data entry from the front panel is outlined in the LogicPak manual and fully described in text when it differs from terminal operation.

3.4 THE FUNCTION MENU

Figure 3-3 shows how the menu will be displayed on your terminal. Table 3-2 summarizes the purpose of each function.

CNTL Z will abort any function on the menu. Escape will take the PLDS out of terminal control. A set of 13 functions and commands exists for data development. Familiarize yourself with them before using the design adapter.

CAUTION

BEFORE REMOVING THE ADAPTER, enter an <ESC> from the terminal; or from the Programmer front panel, press the KEYBOARD key (on the System 19) or the VERIFY key (on the 100A or 29A). The processor in the programmer executes firmware resident in the adapter. These precautions must be taken before removing the adapter from the LogicPak to prevent a program interrupt and resulting loss of RAM data.

3.5 DATA DEVELOPMENT

This section describes the key sequences (fig. 3-1) used for data development. They are arranged in the following order:

1. Input data.
2. Edit data.
3. Inspect the fuse pattern.
4. Enter data to verify and test the device.

Option steps include:

5. Test and Verify options.
6. Download data for documentation or inspection.

Finally:

7. Program the device, (instructions in p/t adapter manual).

Terminal Data Entry If you are developing new data you will need to use the edit mode. If you are updating data previously stored on another media, or entering them after they have been partially developed on a host computer, you will need to use one of the two receive modes.

Data can be loaded into the programmer RAM in one of two formats: the JEDEC format or the ASCII-logic format.

3.5.1 RECEIVE FUSE DATA (B)(EB)(JEDEC FORMAT)

This function prepares the programmer to receive source equations from a peripheral via the serial port in the JEDEC format (see LogicPak manual for details of JEDEC format). The programmer must be prepared to receive data before the transmitting unit begins to send. The key sequence is as follows:

1. Complete the preoperation procedures given in section 3.3. The function menu will be presented.

2. Enter B.

3. The programmer is now ready to receive data. The transmiting unit can now be told to send data.

All line feeds (<LF>) are stripped from the data on input. <CR> is inserted after every 80 characters unless they occur sooner. The data is sequentially stored in the source buffer.

Logic Data can now be displayed by returning to the display menu with a 0, then pressing 4. Normal editing can be performed from this point.

3.5.2 RECEIVE SIGNETICS TRANSLATOR (2)(E2) (ASCII-LOGIC FORMAT)

The ASCII-logic format is only recommended when it is necessary to remain compatible with an existing installation. Otherwise, the JEDEC format is preferred. It allows transmission of test data, and is a more widely accepted format.

The key sequence for entering data in the ASCII-logic format is as follows:

1. Complete the preoperation procedures given in section 3.3. The function menu will be displayed.

2. Enter 2.

3. The programmer is now ready to receive data.

When transmitting logic programming data to or from the programmer's serial port in the ASCII-logic format, you must proceed as described in the following paragraphs.

Figure 3-4 illustrates input files for all five logic devices. Start the stream with an SOH (CNTL A) or an STX (CNTL B).

Header information can be included but it must appear before the SOH or STX. Signal the end of transmission with an ETX (CNTL C).

Logic states for each array are grouped in fields of variables. Each field is preceded by its own ID code: an asterisk and a letter. Tables 3-3 through 3-6 describe the valid states and ID codes for each device.

Term numbers may appear in any order. Within each term, all fields must appear in the order described for each device.

To erase an entry, follow it with a backspace and a rubout. To erase an entire term's entries, enter the ID code, one space, the term number, and the letter E.

NOTE

The programmer has no timeout (normally 25 seconds) when entering data in the E2 input mode. Take your time.

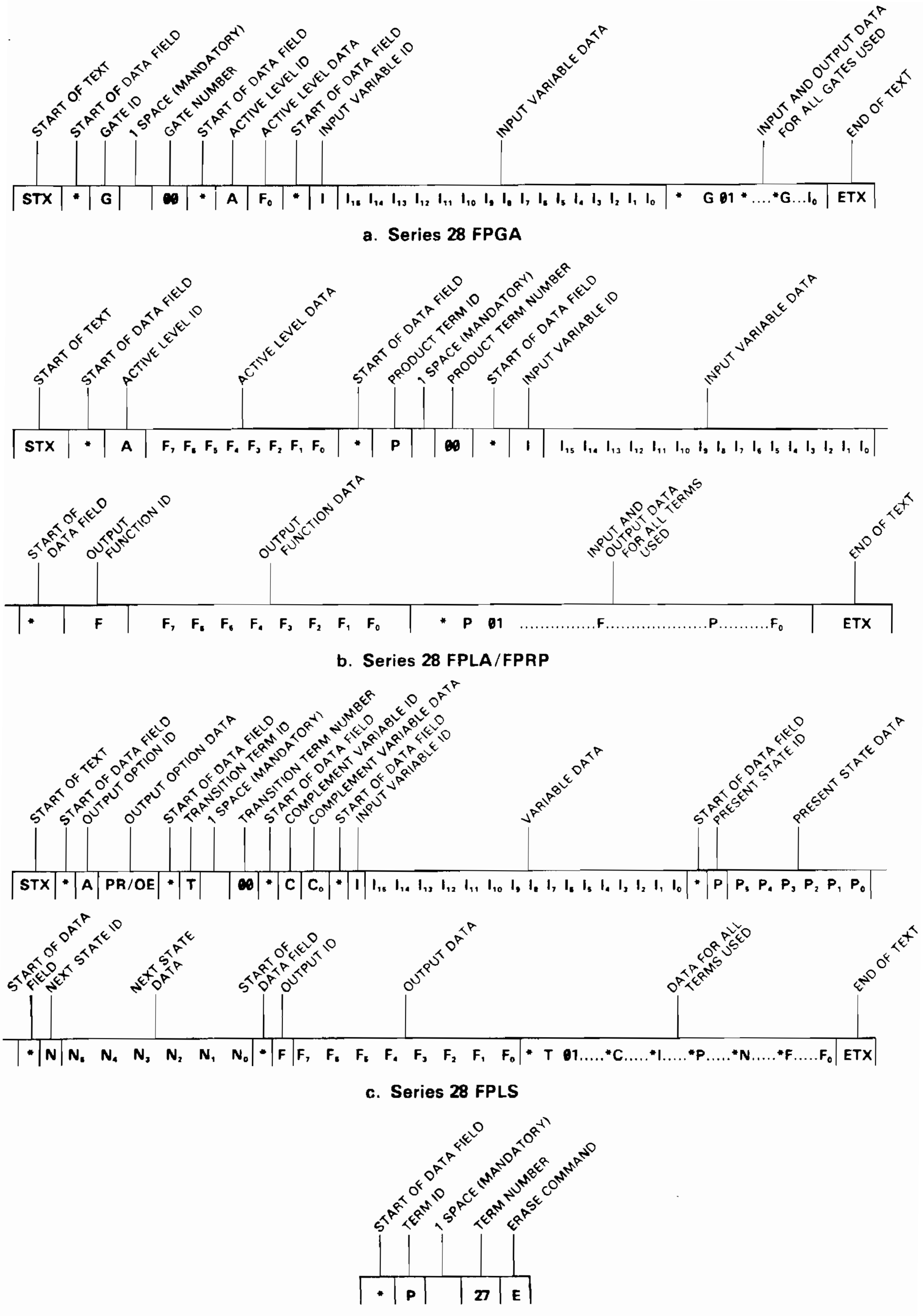
Series 28 FPGA. Table 3-3 lists the valid states and field ID codes for this devices.

Table 3-3. Series 28 FPGA ID Codes and Logic Entry.

	Gate ID = *G (Gate Number)	
Entry	Output Active-Level ID = *A	Input Array ID = *I
H	HIGH	TRUE
L	LOW	FALSE
-	NA	DON'T CARE
Ø	NA	INACTIVE

1. Invoke Select code 2.

2. Enter CNTL A or CNTL B.

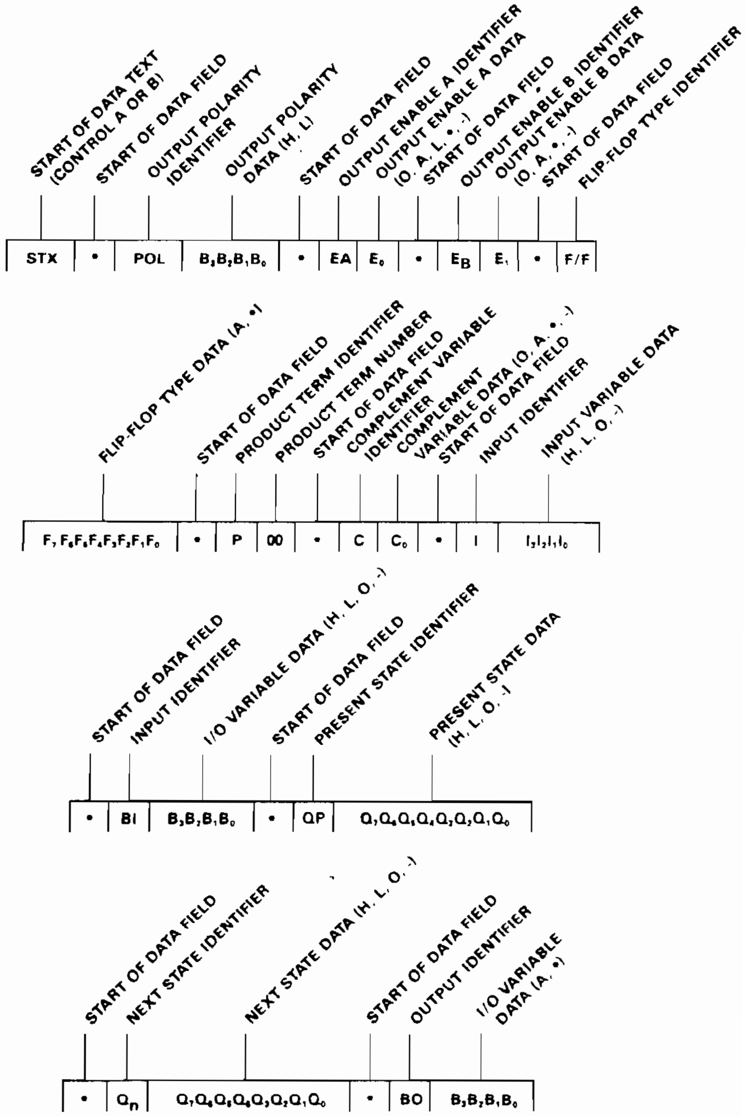


a. Series 28 FPGA

b. Series 28 FPLA/FPRP

c. Series 28 FPLS

Figure 3-4. Input Data Stream Format


START OF DATA TEXT (CONTROL A OR B)
START OF DATA FIELD
OUTPUT POLARITY IDENTIFIER
OUTPUT POLARITY DATA (H, L)
START OF DATA FIELD
OUTPUT ENABLE A IDENTIFIER
OUTPUT ENABLE A DATA (O, A, L, •, -)
START OF DATA FIELD
OUTPUT ENABLE B IDENTIFIER
OUTPUT ENABLE B DATA (O, A, •, -)
START OF DATA FIELD
FLIP-FLOP TYPE IDENTIFIER
STX
•
POL
$B_3B_2B_1B_0$
•
EA
E_0
•
E_B
E_1
•
F/F
FLIP-FLOP TYPE DATA (A, •)
START OF DATA FIELD
PRODUCT TERM IDENTIFIER
PRODUCT TERM NUMBER
START OF DATA FIELD
COMPLEMENT VARIABLE IDENTIFIER
COMPLEMENT VARIABLE DATA (O, A, •, -)
START OF DATA FIELD
INPUT IDENTIFIER
INPUT VARIABLE DATA (H, L, O, -)
$F_7F_6F_5F_4F_3F_2F_1F_0$
•
P
00
•
C
C_0
•
I
$I_3I_2I_1I_0$
START OF DATA FIELD
INPUT IDENTIFIER
I/O VARIABLE DATA (H, L, O, -)
START OF DATA FIELD
PRESENT STATE IDENTIFIER
PRESENT STATE DATA (H, L, O, -)
•
BI
$B_3B_2B_1B_0$
•
QP
$Q_7Q_6Q_5Q_4Q_3Q_2Q_1Q_0$
START OF DATA FIELD
NEXT STATE IDENTIFIER
NEXT STATE DATA (H, L, O, -)
START OF DATA FIELD
OUTPUT IDENTIFIER
I/O VARIABLE DATA (A, •)
•
Q_n
$Q_7Q_6Q_5Q_4Q_3Q_2Q_1Q_0$
•
BO
$B_3B_2B_1B_0$

e. Series 20 FPLS

Note: The last line is not valid for control terms

Figure 3-4. Input Data Stream Format (Continued)

3. Enter *G, exactly one space, and a gate number. Gates do not have to appear in numerical order, but each gate must include both fields of variables (steps 4 through 7).

4. Enter *A.

5. Enter the output active level for this gate.

6. Enter *I.

7. Enter all 16 of the input array variables starting with the highest numbered input.

8. Repeat steps 3 through 7 until you've entered all of your function table.

9. Enter CNTL C.

10. Enter escape. Change to p/t adapter.

11. Insert a blank device in the socket and start a Program operation.

Series 28 FPLA/FPRP. Table 3-4 lists the valid states and field ID codes for this device.

Table 3-4. Series 28 FPLA/FPRP ID Codes and Logic Entries

ENTRY	FPLA OUTPUT ACTIVE LEVEL ID = *A	TERM ID = * (TERM NUMBER)	
		INPUT ARRAY ID = *1	OUPUT ARRAY ID = F*
H	HIGH	TRUE	NA
L	LOW	FALSE	NA
-	NA	DON'T CARE	NA
Ø	NA	INACTIVE	NA
A	NA	NA	ACTIVE
0	NA	NA	INACTIVE

1. Invoke Select Code 2.
2. Enter CNTL A or CNTL B.
3. Enter *A.
4. Enter states for all eight of the output active-level array variables starting with the highest numbered output.

NOTE

For programming a series 28 FPRP, enter H (active high) for all of the output active levels.

5. Enter *P, exactly one space, and a product term number. Product terms do not have to appear in numerical order, but each must include both fields of variables (steps 6 through 9).

6. Enter *I.

7. Enter states for all 16 input array variables, starting with the highest numbered input.

8. Enter *F.

9. Enter the states for all eight of the output array variables, starting with the highest numbered output.

10. Repeat steps 5 through 9 until you've entered all of your function tables.

11. Enter CNTL C.

12. Enter escape. Change to p/t adapter.

13. Insert a blank device in the socket adapter and start a Program operation.

Series 28 FPLS. Table 3-5 lists the valid state entries and field ID codes for 28 series input.

1. Invoke Select code 2.

2. Enter CNTL A or CNTL B.

3. Enter *A.

4. Enter a character to set the PR/OE option to its desired state.

5. Enter *T, exactly one space and a transition term number. Transition terms do not have to appear in numerical order, but each must include all five fields of variables (steps 6 through 15).

6. Enter *C.

7. Enter a state for the complement array variable.

8. Enter *I.

9. Enter states for all 16 of the input array variables with the highest numbered input first.

10. Enter *P.

Table 3-5. Series 28 FPLS ID Codes and Logic Arrays

ENTRY	PR/OE ID = *A	TERM ID = *T (TERM NUMBER) COMPLEMENT ARRAY ID = *C	INPUT & PRESET STATE ARRAYS ID = *1 & *P	NEXT STATE & OUTPUT ARRAYS ID = *N & &F
H	PRESET	NA	TRUE	SET
L	OUTPUT ENABLE	NA	FALSE	RESET
-	NA	TRANSPARENT	DON'T CARE	NO CHANGE
Ø	NA	INACTIVE	INACTIVE	INACTIVE
A	NA	GENERATE	NA	NA
O	NA	PROPAGATE	NA	NA

11. Enter states for all six of the present state arrays variables with the highest numbered variable first.

12. Enter *N.

13. Enter states for all six of the next state array variables with the highest numbered variable appearing first.

14. Enter *F.

15. Enter states for all six of the next state array variables with the highest numbered variable appearing first.

16. Repeat steps 5 through 15 until you've entered the entire function table.

17. Enter CNTL C.

18. Enter escape. Change to p/t adapter.

19. Insert a blank device in the socket adapter and start a Program operation.

Series 20 FPLA. Table 3-6 lists the valid entries for the series 20 FPLA input.

Table 3-6. Series 20 FPLA Logic Entry

CHARACTER	ACTIVE LEVEL	I ARRAY	B(I) ARRAY	B(O) ARRAY
H	HIGH	TRUE	TRUE	N/A
L	LOW	FALSE	FALSE	FALSE
-	NA	DON'T CARE	DON'T CARE	NA
Ø	NA	INACTIVE	INACTIVE	NA
A	NA	NA	NA	ACTIVE
o	NA	NA	NA	INACTIVE

1. Invoke Select code 2.

2. Enter CNTL A or CNTL B.

3. Enter * POL.

4. Enter states for all ten of the output-polarity array variables starting with the highest numbered output.

5. Enter *P and a product term number. Product terms do not have to appear in numerical order, but each product term must include all the fields of variables (steps 6 through 11).

6. Enter *I.

7. Enter states for all eight input array variables, starting with the highest numbered input.

8. Enter *BI.

9. Enter states for all ten B(I) input array variables, starting with the highest numbered input.

10. Enter *BO (only for terms Ø through 31).

11. Enter states for all ten B(0) output array variables, starting with the highest numbered output (only for terms Ø through 31).

12. Repeat steps 5 through 11 until you have entered all of your function table.

13. Enter CNTL C.

14. Enter escape. Change to p/t adapter.

15. Insert a blank device in the socket adapter and start a Program.

Series 20 FPLS. Table 3-7 lists the valid entries for the series 20 FPLS input.

1. Invoke Select Code 2.

2. Enter CNTL A or CNTL B.

3. Enter * POL.

4. Enter states for all four output polarity array variables starting with the highest numbered output.

5. Enter * EA.

6. Enter states for the output enable variable.

7. Enter * EB.

8. Enter states for the output enable variable B.

9. Enter * F/F.

10. Enter states for all eight flip-flop function variables starting with the highest numbered output.

11. Enter * P and a product term number or letters in the case of the control terms. Product terms do not have to appear in numberical order, but each product term must include all the fields of variables (Steps 12 - 23).

12. Enter * C.

13. Enter state for the complement array variable.

14. Enter * I.

15. Enter states for all four input variables.

16. Enter * BI.

17. Enter states for all four static I/O variables.

18. Enter * QP.

19. Enter states for all eight registered feedback variables.

NOTE: Steps 20 - 23 do not apply for control terms.

20. Enter * QN

21. Enter states for all eight registered OR array variables.

22. Enter * BO.

23. Enter states for all four OR array variable for the static outputs.

24. Repeat steps 12 - 23 for the entire function table.

25. Enter CNTL C.

26. Enter escape. Change to P/T adapter.

27. Insert a blank device in the socket adapter and program.

Table 3-7. Series 20 FPLS Logic Entry

	Active	EA/EB	F/F	C	I	B(I)	Q(P)	Q(N)*	B(O)*
Character	Level	Option	Option	Term	Array	Array	Array	Array	Array
H	High	-	NA	NA	True	True	True	True	NA
L	Low	-	NA	NA	False	False	False	False	NA
0	NA	-	NA	Inactive	Inactive	Inactive	Inactive	Inactive	NA
-	NA	-	NA	Transport	Don't care	Don't care	Don't care	Don't care	NA
A	NA	-	Controlled JK/D	Generate	N/A	N/A	N/A	N/A	Active
o	NA	-	J-K	Propagate	N/A	N/A	N/A	N/A	Inactive

*These two fields do not exist for the control terms

3.5.3 EDIT LOGIC DIAGRAM (4)

This function is used to enter and edit data. Figure 3-5 is a flowchart of the editing process. Table 3-8 lists the commands you can use to manipulate logic entries, control the display, and move the cursor.

To enter data using the terminal entry mode, enter 4.

The display will look like one of the title block figure 3-9, depending on the device selected, unless no device type or Family and Pinout Codes have been entered.

You are presented with a series of fields corresponding to the arrays within a device you will later program. To load RAM with your function table, enter logic state characters in these fields. The initial display includes the first field of the device.

The following paragraphs explain how to enter logic states and how to edit data after it has been entered.

Series 28 FPGA. The terminal display will look like figure 3-9a. Table 3-3 lists the valid logic state entries for the series 28 FPGA.

1. Enter a gate number. (Remember to enter gate number 5 as Ø5, gate number 2 as Ø2, and so forth). "N" will advance the cursor to the next field. The cursor will then move to the single variable for the output active-level field.

2. Enter a state for the output active level.. The cursor will then move to the first variable of the input array field.

3. Enter states for the input array variables.

4. When you finish a gate's variables, the next gate's variables will appear and the cursor will return to the active-level variable. Repeat steps 2 and 3 for each gate, until you have loaded all of the function table.

Series 28 FPLA/FPRP. The terminal display will look like figure 3-9b. Table 3-4 lists the logic entries to use for these devices.

1. Series 28FPLA only: Enter the active levels for the product term outputs. For series 28 FPRP, enter H (active high) for all of the output-active levels.

2. Next, you'll see the two product term fields displayed beginning with product term ØØ as in figure 3-10.

3. Enter the number of the product term you want to work on. The cursor will then be on the first variable of the input array field.

4. Enter logic states for the input array variables.

5. After setting the last variable in the field, the cursor will jump to the next field, the output array variables.

6. When you finish the term, the cursor will return to the first variable of the input array field, and the product term number will increment. Repeat steps 4 and 5 until you've entered all of your function.

Series 28 FPLS. The terminal display will look like figure 3-9c. Table 3-5 lists the valid logic states for the Series 28 FPLS.

1. Enter the PR/OE option variable.

2. Next, the 5 fields of the transition terms will appear on the screen as in figure 3-11.

3. Enter the number of the transition term you want to work on. The cursor will then be on the variable in the complement array field.

4. Enter a state for the complement array. The cursor will then move to the first variable of the next field.

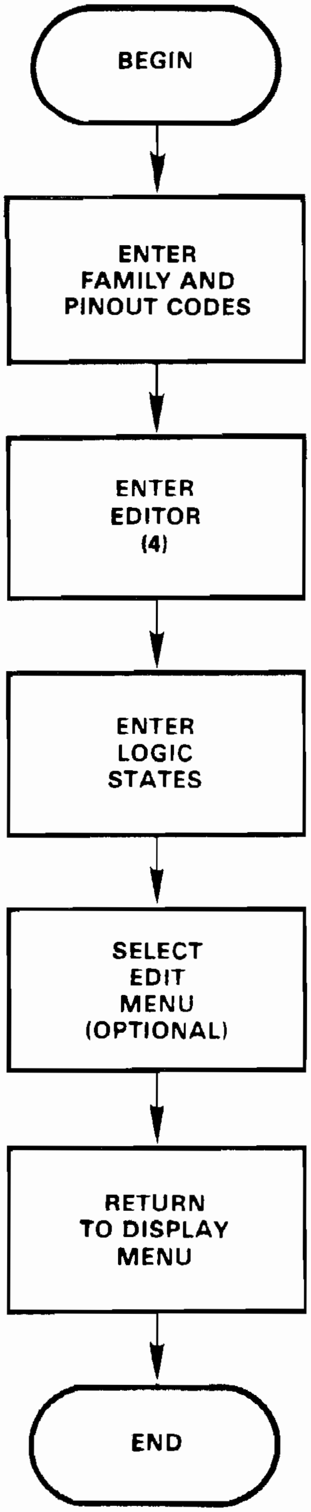
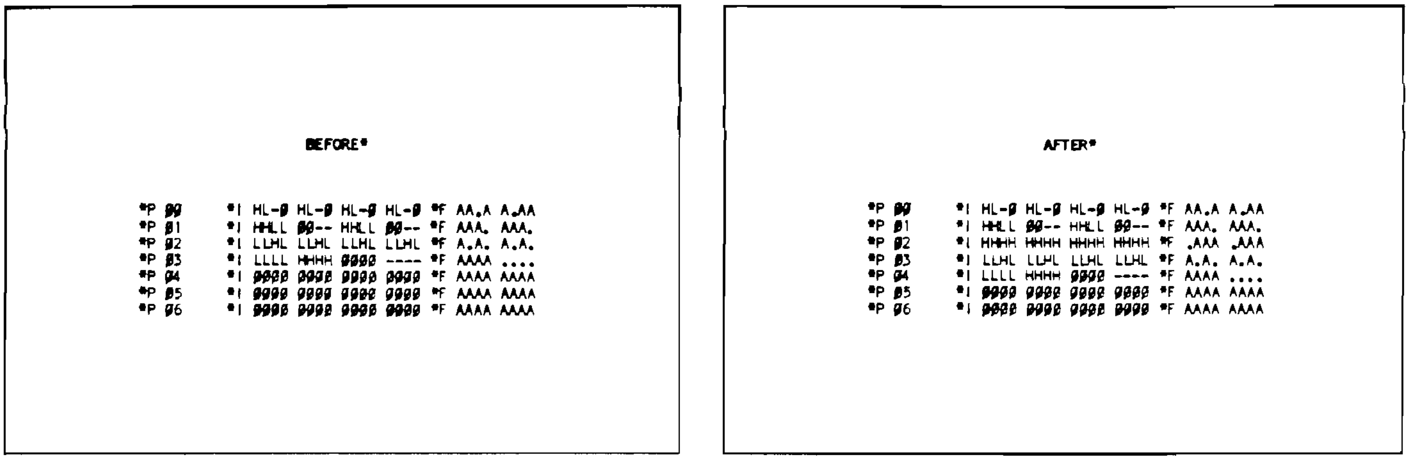


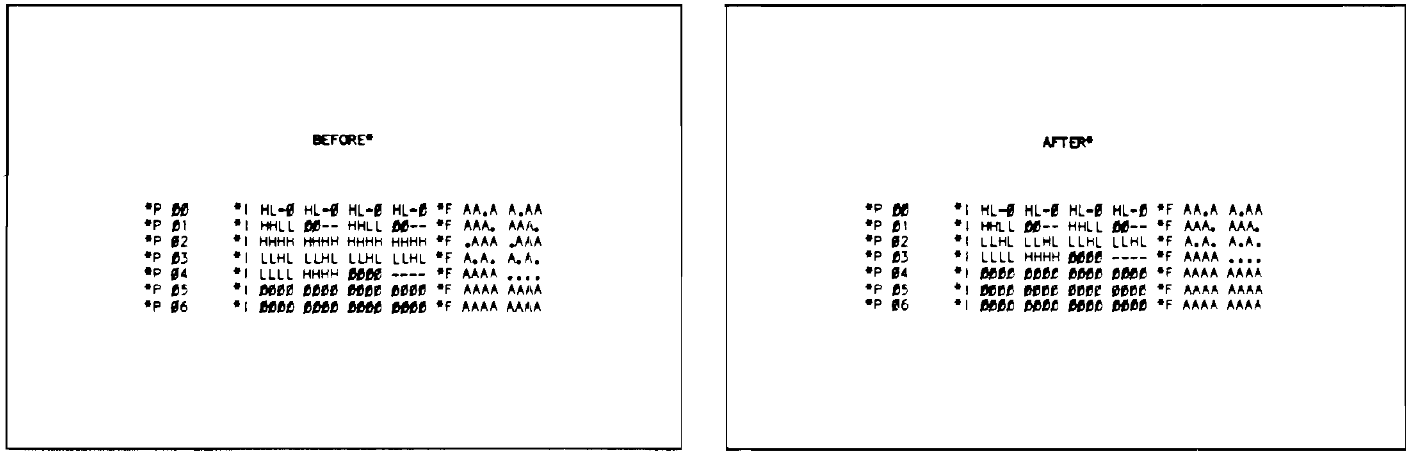
Figure 3-5. Flowchart, Editing Process



```
BEFORE*

*P ØØ     *I HL-Ø HL-Ø HL-Ø HL-Ø *F AA.A A.AA
*P Ø1     *I HHLL ØØ-- HHLL ØØ-- *F AAA. AAA.
*P Ø2     *I LLHL LLHL LLHL LLHL *F A.A. A.A.
*P Ø3     *I LLLL HHHH ØØØØ ---- *F AAAA ....
*P Ø4     *I ØØØØ ØØØØ ØØØØ ØØØØ *F AAAA AAAA
*P Ø5     *I ØØØØ ØØØØ ØØØØ ØØØØ *F AAAA AAAA
*P Ø6     *I ØØØØ ØØØØ ØØØØ ØØØØ *F AAAA AAAA
```

```
AFTER*

*P ØØ     *I HL-Ø HL-Ø HL-Ø HL-Ø *F AA.A A.AA
*P Ø1     *I HHLL ØØ-- HHLL ØØ-- *F AAA. AAA.
*P Ø2     *I HHHH HHHH HHHH HHHH *F .AAA .AAA
*P Ø3     *I LLHL LLHL LLHL LLHL *F A.A. A.A.
*P Ø4     *I LLLL HHHH ØØØØ ---- *F AAAA ....
*P Ø5     *I ØØØØ ØØØØ ØØØØ ØØØØ *F AAAA AAAA
*P Ø6     *I ØØØØ ØØØØ ØØØØ ØØØØ *F AAAA AAAA
```

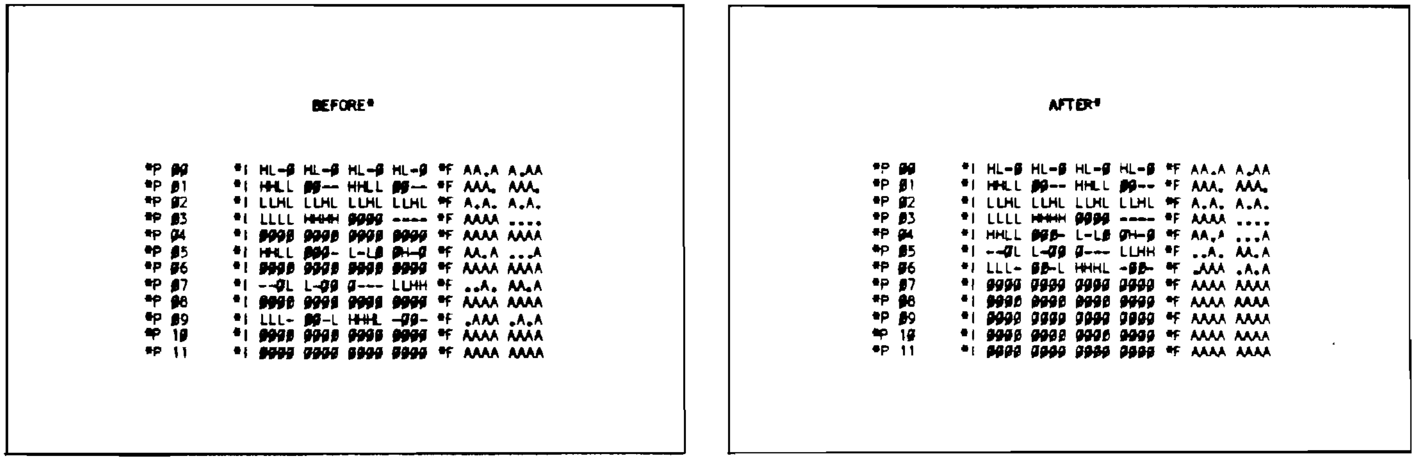
Figure 3-6. Insert Series 28 FPLA Data



```
BEFORE*

*P ØØ     *I HL-Ø HL-Ø HL-Ø HL-Ø *F AA.A A.AA
*P Ø1     *I HHLL ØØ-- HHLL ØØ-- *F AAA. AAA.
*P Ø2     *I HHHH HHHH HHHH HHHH *F .AAA .AAA
*P Ø3     *I LLHL LLHL LLHL LLHL *F A.A. A.A.
*P Ø4     *I LLLL HHHH ØØØØ ---- *F AAAA ....
*P Ø5     *I ØØØØ ØØØØ ØØØØ ØØØØ *F AAAA AAAA
*P Ø6     *I ØØØØ ØØØØ ØØØØ ØØØØ *F AAAA AAAA
```

```
AFTER*

*P ØØ     *I HL-Ø HL-Ø HL-Ø HL-Ø *F AA.A A.AA
*P Ø1     *I HHLL ØØ-- HHLL ØØ-- *F AAA. AAA.
*P Ø2     *I LLHL LLHL LLHL LLHL *F A.A. A.A.
*P Ø3     *I LLLL HHHH ØØØØ ---- *F AAAA ....
*P Ø4     *I ØØØØ ØØØØ ØØØØ ØØØØ *F AAAA AAAA
*P Ø5     *I ØØØØ ØØØØ ØØØØ ØØØØ *F AAAA AAAA
*P Ø6     *I ØØØØ ØØØØ ØØØØ ØØØØ *F AAAA AAAA
```

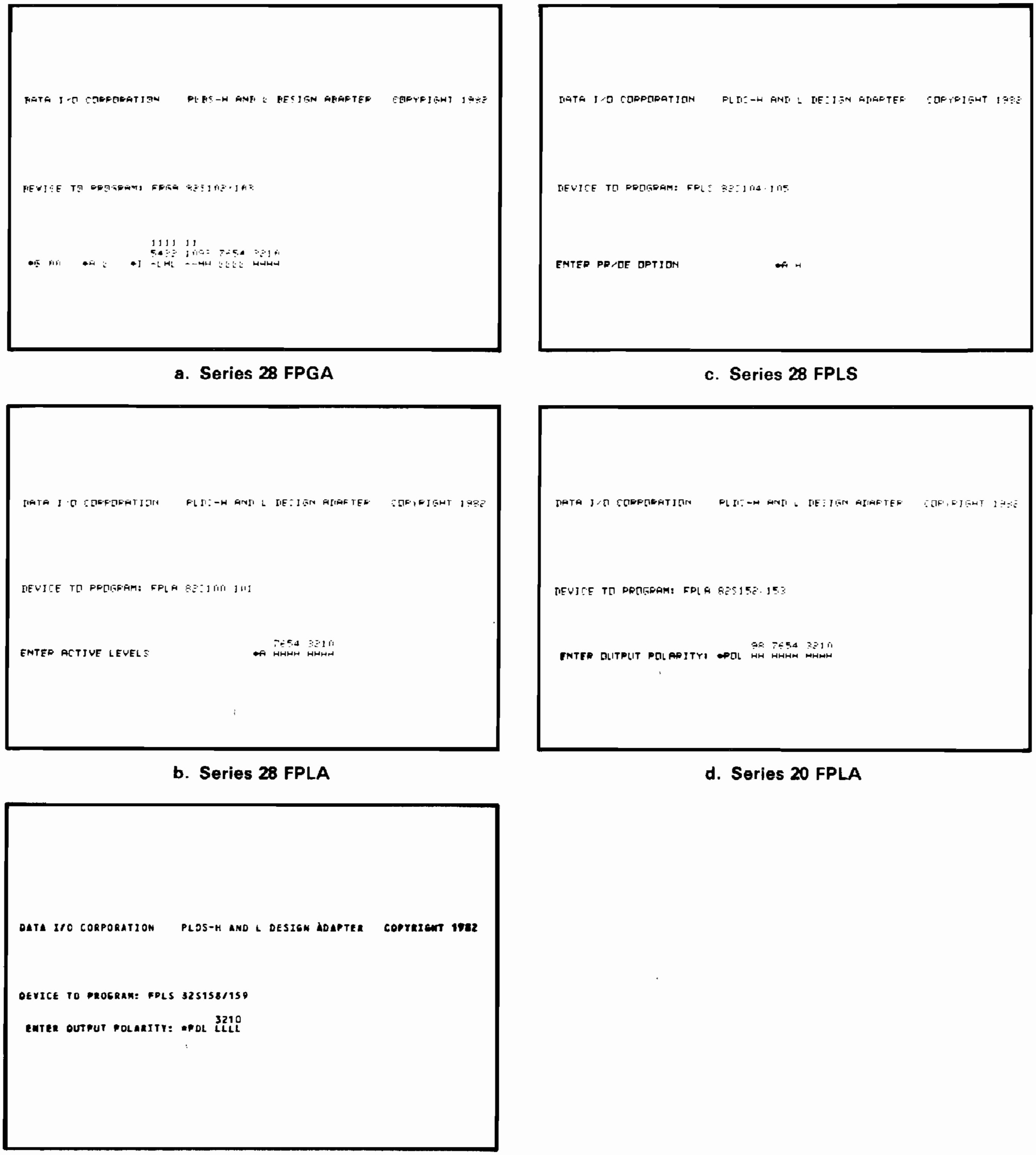
Figure 3-7. Delete Series 28 FPLA Data



```
BEFORE*

*P ØØ     *I HL-Ø HL-Ø HL-Ø HL-Ø *F AA.A A.AA
*P Ø1     *I HHLL ØØ-- HHLL ØØ-- *F AAA. AAA.
*P Ø2     *I LLHL LLHL LLHL LLHL *F A.A. A.A.
*P Ø3     *I LLLL HHHH ØØØØ ---- *F AAAA ....
*P Ø4     *I ØØØØ ØØØØ ØØØØ ØØØØ *F AAAA AAAA
*P Ø5     *I HHLL ØØØ- L-LØ ØH-Ø *F AA.A ...A
*P Ø6     *I ØØØØ ØØØØ ØØØØ ØØØØ *F AAAA AAAA
*P Ø7     *I --ØL L-ØØ Ø--- LLHH *F ..A. AA.A
*P Ø8     *I ØØØØ ØØØØ ØØØØ ØØØØ *F AAAA AAAA
*P Ø9     *I LLL- ØØ-L HHHL -ØØ- *F .AAA .A.A
*P 1Ø     *I ØØØØ ØØØØ ØØØØ ØØØØ *F AAAA AAAA
*P 11     *I ØØØØ ØØØØ ØØØØ ØØØØ *F AAAA AAAA
```

```
AFTER*

*P ØØ     *I HL-Ø HL-Ø HL-Ø HL-Ø *F AA.A A.AA
*P Ø1     *I HHLL ØØ-- HHLL ØØ-- *F AAA. AAA.
*P Ø2     *I LLHL LLHL LLHL LLHL *F A.A. A.A.
*P Ø3     *I LLLL HHHH ØØØØ ---- *F AAAA ....
*P Ø4     *I HHLL ØØØ- L-LØ ØH-Ø *F AA.A ...A
*P Ø5     *I --ØL L-ØØ Ø--- LLHH *F ..A. AA.A
*P Ø6     *I LLL- ØØ-L HHHL -ØØ- *F .AAA .A.A
*P Ø7     *I ØØØØ ØØØØ ØØØØ ØØØØ *F AAAA AAAA
*P Ø8     *I ØØØØ ØØØØ ØØØØ ØØØØ *F AAAA AAAA
*P Ø9     *I ØØØØ ØØØØ ØØØØ ØØØØ *F AAAA AAAA
*P 1Ø     *I ØØØØ ØØØØ ØØØØ ØØØØ *F AAAA AAAA
*P 11     *I ØØØØ ØØØØ ØØØØ ØØØØ *F AAAA AAAA
```

Figure 3-8. Compress Series 28 FPLA Terms



```
DATA I/O CORPORATION    PLDS-H AND L DESIGN ADAPTER    COPYRIGHT 1982


DEVICE TO PROGRAM: FPGA 82S102/103


                     1111 11
                     5432 1098 7654 3210
 *S 00    *A L    *I -LHL --HH LLLL HHHH
```

a. Series 28 FPGA

```
DATA I/O CORPORATION    PLDS-H AND L DESIGN ADAPTER    COPYRIGHT 1982


DEVICE TO PROGRAM: FPLS 82S104/105


ENTER PR/OE OPTION                *A H
```

c. Series 28 FPLS

```
DATA I/O CORPORATION    PLDS-H AND L DESIGN ADAPTER    COPYRIGHT 1982


DEVICE TO PROGRAM: FPLA 82S100/101


                                 7654 3210
ENTER ACTIVE LEVELS            *A HHHH HHHH
```

b. Series 28 FPLA

```
DATA I/O CORPORATION    PLDS-H AND L DESIGN ADAPTER    COPYRIGHT 1982


DEVICE TO PROGRAM: FPLA 82S152/153

                              98 7654 3210
 ENTER OUTPUT POLARITY: *POL HH HHHH HHHH
```

d. Series 20 FPLA

```
DATA I/O CORPORATION    PLDS-H AND L DESIGN ADAPTER    COPYRIGHT 1982


DEVICE TO PROGRAM: FPLS 82S158/159

                             3210
 ENTER OUTPUT POLARITY: *POL LLLL
```

e. Series 20 FPLS

Figure 3-9. Editing Title Blocks for IFL Devices (Terminal Entry Mode)

5. Enter states for the variables in the input array field. The cursor will then move to the first variable in the next field.

6. Enter characters for the present-state array field. The cursor will then jump to the first variable of the next field.

7. Enter logic states for the next-state array field. The cursor will then move to the first variable in the last field.

8. Enter logic states for the output array field. When you finish, the cursor will return to the variable in the complement array field, and the transition term number will increment.

9. Repeats steps 4 through 8 until you've entered your entire function table.

Series 20 FPLA. The terminal display will look like figure 3-9d. Table 3-6 lists the logic entries to use for the Series 20 FPLA.

1. Enter the active levels for the product term outputs.

2. Next, you will see the three product term fields displayed, as in figure 3-12a.

3. Enter the number of the product term you want to work on. The cursor will then be on the first variable of the input array field.

4. Enter the logic states for the input array variables.

5. After setting the last variable of the field, the cursor will jump to the next field, the BI array variables.

6. Enter the logic states for the BI array variables.

7. Complete the term by entering the logic states of the last field, the BO array variables. Upon completion, the cursor will return to the first variable of the input array field, and the product term will increment.

8. Repeat steps 4 through 7 until you have entered all of the product terms (P_{00} through P_{31}).

9. Upon completion of the last product term (P_{31}), the cursor will return to the first variable in the input array field of the first control term (PD_9) as shown in figure 3-12b.

10. Repeat steps 4 through 6 until you have entered in all of your control terms (PD_9 through PD_0).

Series 20 FPLS. The terminal display will look like Figure 3-9e. Table 3-7 lists the logic entries for the series 20 FPLS.

1. Enter active level states

2. Enter EA/EB options

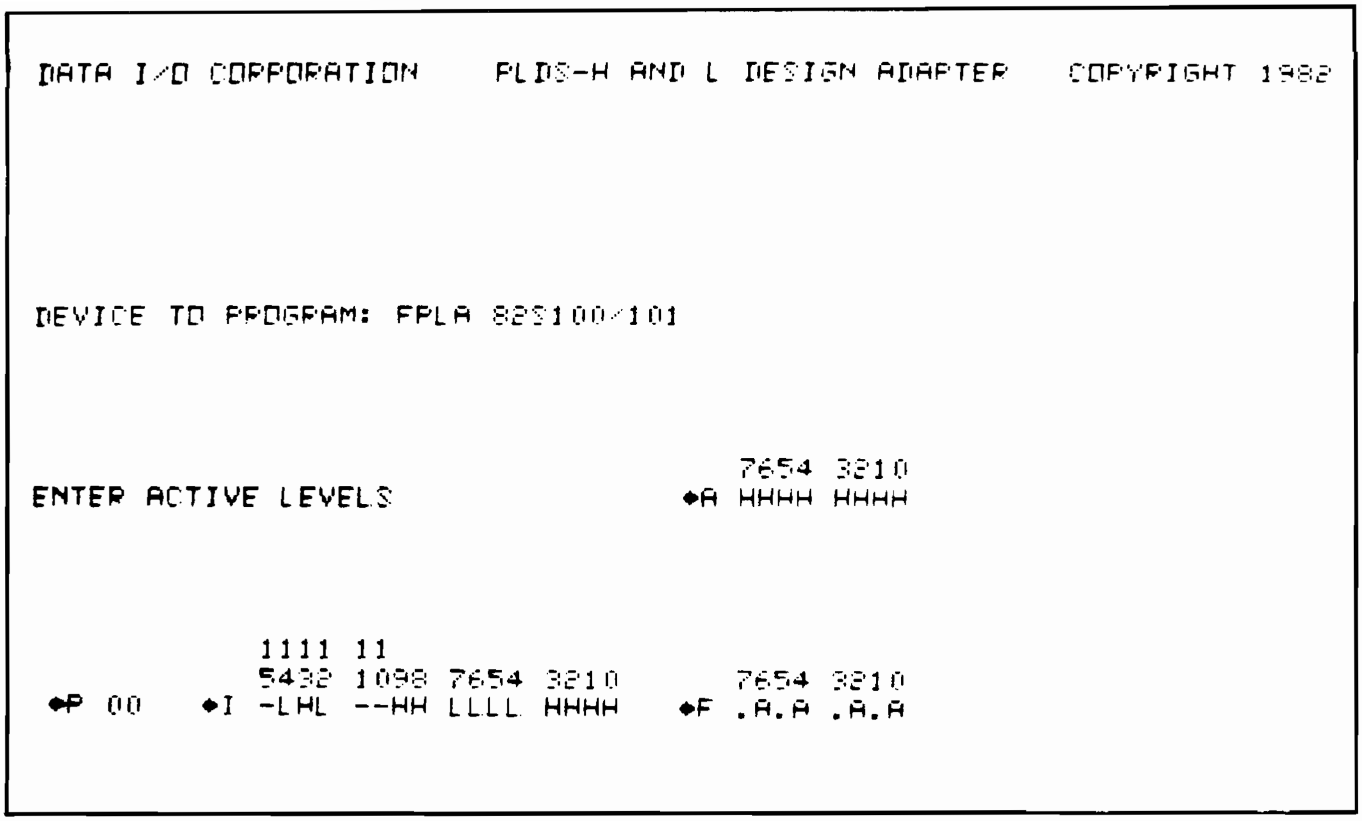
3. Enter F/F type options

4. Next you will see six product term fields displayed. (Control terms will have only four fields).

5. Enter the number, or letters in the case of a control term, of the product term you wish to edit.

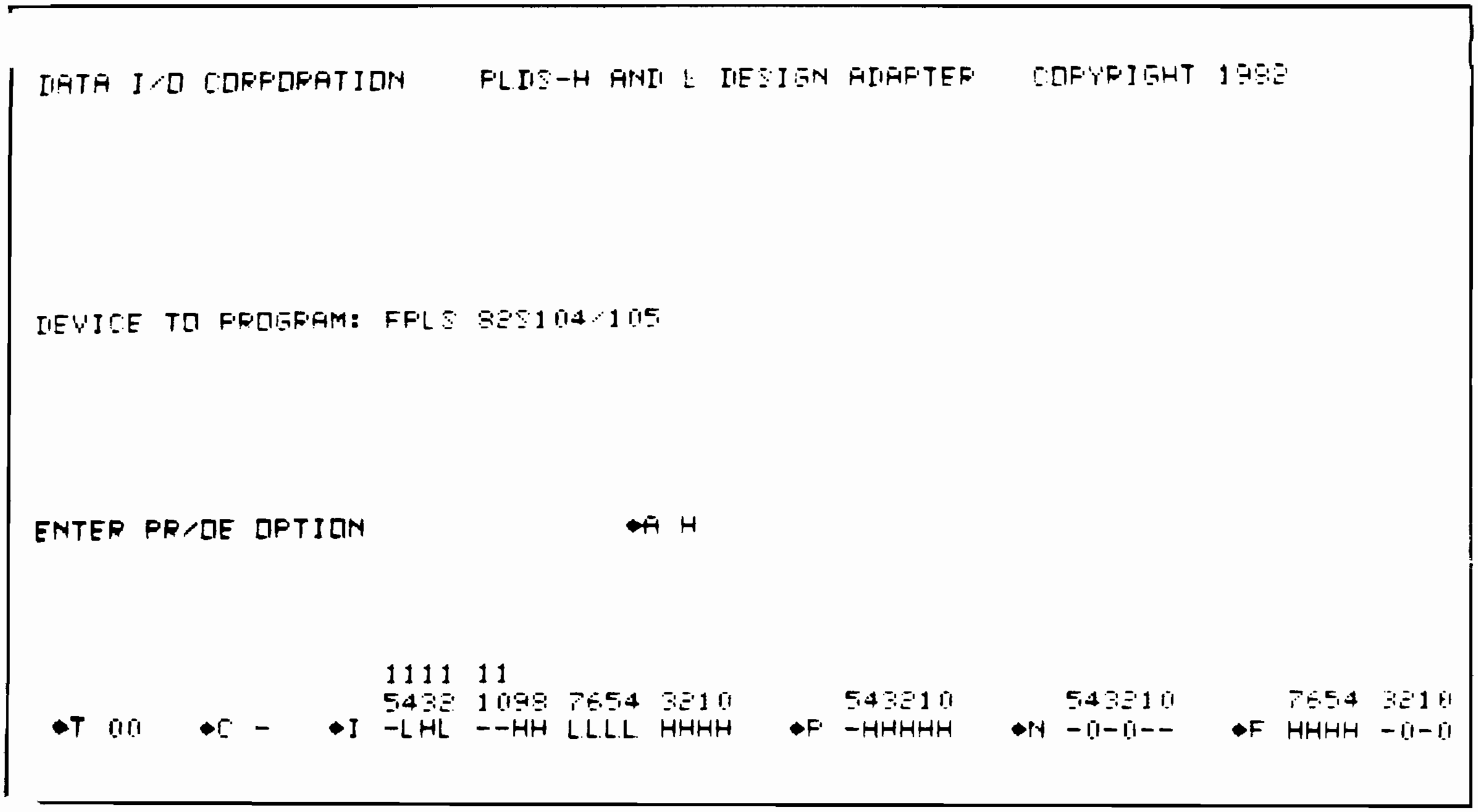
6. Enter states for each field. When a field is completed the cursor will automatically jump to the next field. When finished with a product term the cursor will jump to the beginning of the next product term.

7. Repeat above sequence for all product terms desired to be edited.



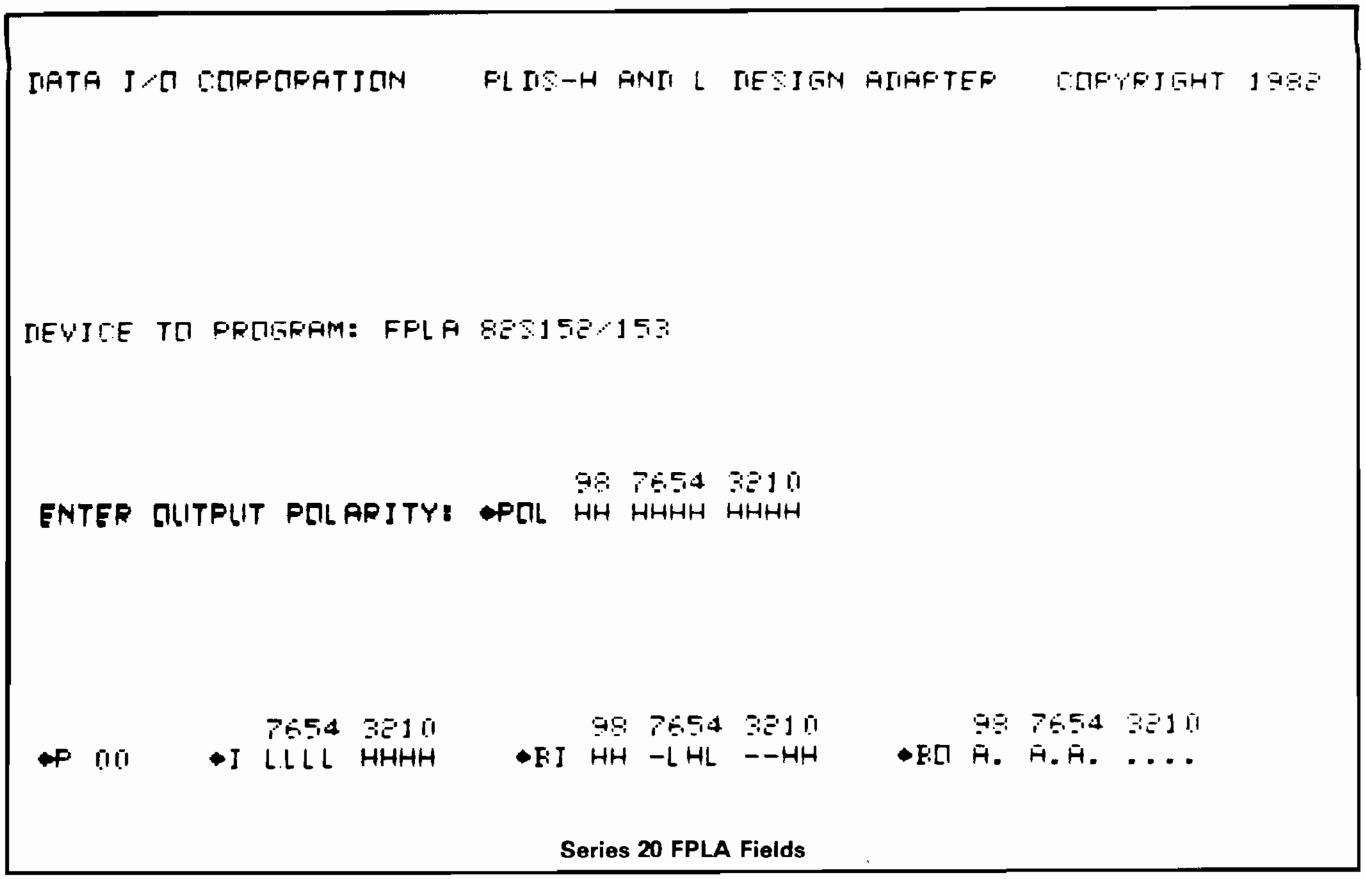
```
DATA I/O CORPORATION    PLDS-H AND L DESIGN ADAPTER    COPYRIGHT 1982



DEVICE TO PROGRAM: FPLA 82S100/101



                                                7654 3210
ENTER ACTIVE LEVELS                          ◆A HHHH HHHH



                1111 11
                5432 1098 7654 3210             7654 3210
 ◆P 00     ◆I -LHL --HH LLLL HHHH          ◆F .A.A .A.A
```

Figure 3-10. Series 28 FPLA/FPRP Title Block and Product Term 00 After Pressing P



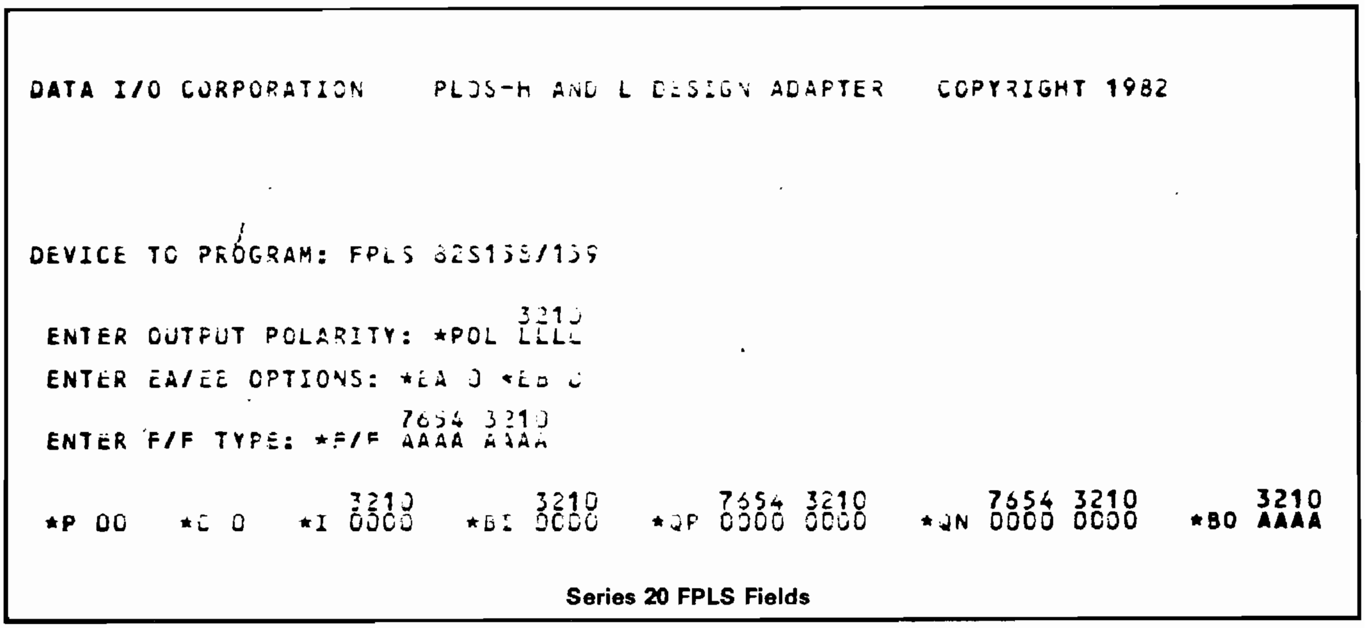
```
DATA I/O CORPORATION    PLDS-H AND L DESIGN ADAPTER   COPYRIGHT 1982



DEVICE TO PROGRAM: FPLS 82S104/105



ENTER PR/OE OPTION                    ◆A H


                       1111 11
                       5432 1098 7654 3210       543210       543210       7654 3210
 ◆T 00     ◆C -     ◆I -LHL --HH LLLL HHHH    ◆P -HHHHH    ◆N -0-0--    ◆F HHHH -0-0
```

Figure 3-11. Series 28 FPLS Title Block and Transition Term 00 After Pressing T



```
DATA I/O CORPORATION    PLDS-H AND L DESIGN ADAPTER   COPYRIGHT 1982




DEVICE TO PROGRAM: FPLA 82S152/153


                                   98 7654 3210
 ENTER OUTPUT POLARITY: ◆POL HH HHHH HHHH




                7654 3210          98 7654 3210          98 7654 3210
 ◆P 00     ◆I LLLL HHHH      ◆BI HH -LHL --HH      ◆BO A. A.A. ....
```

Series 20 FPLA Fields



```
DATA I/O CORPORATION    PLDS-H AND L DESIGN ADAPTER   COPYRIGHT 1982



DEVICE TO PROGRAM: FPLS 82S158/159

                                3210
 ENTER OUTPUT POLARITY: *POL LLLL
 ENTER EA/EB OPTIONS: *EA 0 *EB 0
                          7654 3210
 ENTER F/F TYPE: *F/F AAAA AAAA

                   3210       3210       7654 3210       7654 3210      3210
 *P 00   *C 0   *I 0000   *BI 0000   *QP 0000 0000   *QN 0000 0000   *BO AAAA
```

Series 20 FPLS Fields

Figure 3-12. Series 20 FPLA and FPLS Fields

Table 3-8. Terminal Editing Commands

COMMAND NAME	DESCRIPTION
G Enter Gate Number	Cursor waits for entry of an FPGA gate number.
P Enter Product Term Number	Cursor waits for an entry of an FPLA/FPRP product term number.
T Enter Transition Term Number	Cursor waits for entry of an FPLS transition term number.
> Advance Cursor	Moves cursor to the right.
< Back up Cursor	Moves cursor to the left. Cannot return to previous field.
F Display Next Term	Display next higher-numbered gate or term.
R Display Last Term	Display next lower-numbered gate or term.
N Enter Next Field	Cursor advances to the next field of variables in the current gate or term.
I Insert Term	Inserts new term before term display when command. The inserted displayed term will be displayed and will show its default (unprogrammed) values; new states can then be entered. Higher-numbered terms move up one term. Highest device term is lost. See figure 3-6. (takes several seconds to execute).
D Delete Term	Data at displayed term is destroyed. Data at higher-numbered terms move down one term. See figure 3-7.
C Clear Term	Displayed term is cleared to its default values.
X Deactivate Term	Outputs of displayed term are set to inactive. FPLA and FPLS only.
E Display Edit Menu	Displays edit menu.
0 Exit	Returns to display menu.
1 Terminal Entry	Returns to Edit Mode.
2 Serial Input	Receive data in ASCII-logic format.
3 Serial Output	Transmit Data in ASCII-logic format.
4 List Low-Order Terms	Terms 0-23 are displayed.
5 List High-Order Terms	Terms 24-47 are displayed.
6 Compress Terms	Nonconsecutive terms are moved up to occupy intervening blank terms. See figure 3-8. The screen will display nonverifying data: RAM data wil appear first, with corresponding device data on the line below.
CNTL Z Escape Logic Edit Mode	CNTL Z causes a return to the display menu.
CNTL S Halt Transmission	Use this command to interrupt serial output or verify output.
CNTL Q Resume Transmission	Use this command to resume transmission after interrupting serial output or verify output.

After all variables have been entered into the programmer RAM, check the fuse table against the logic diagram to ensure that all entries were correctly made. This can be accomplished with the Display Fuse Pattern function.

3.5.4 DISPLAY FUSE PATTERN (A).

By comparing the logic diagram with the fuse pattern (fig. 3-13 and fig. 3-14), you can see that the fuse numbers correspond to those on the logic diagram. The fuse table matrix usually represents the same physical layout as the logic diagram. Fuse numbers have been added to make this comparison easier.

To display the fuse pattern of a device enter "A" from the terminal display menu. The pattern will scroll from start to finish on a CRT terminal. To Halt the display, enter CNTL S. To continue the scroll, enter CNTL Q. The fuse pattern will be a result of either input from a peripheral or Logic data in the edit mode. If it is necessary to enter decimal fuse data and alter specific fuse states, this can be done at this time with function E from the edit menu. Data I/O does not recommend this procedure, however, because (1) the resulting fuse pattern will no longer represent your function table and (2) this will negate any Structured Test vectors that were derived from your function table.

FIELD PROGRAMMABLE GATE ARRAY (16X9) 82S102 (O.C.)/82S103 (T.S.)

INTEGRATED FUSE LOGIC
SERIES 28

= 0 = Fuse Intact
= 1 = Fuse Blown

FPGA LOGIC DIAGRAM

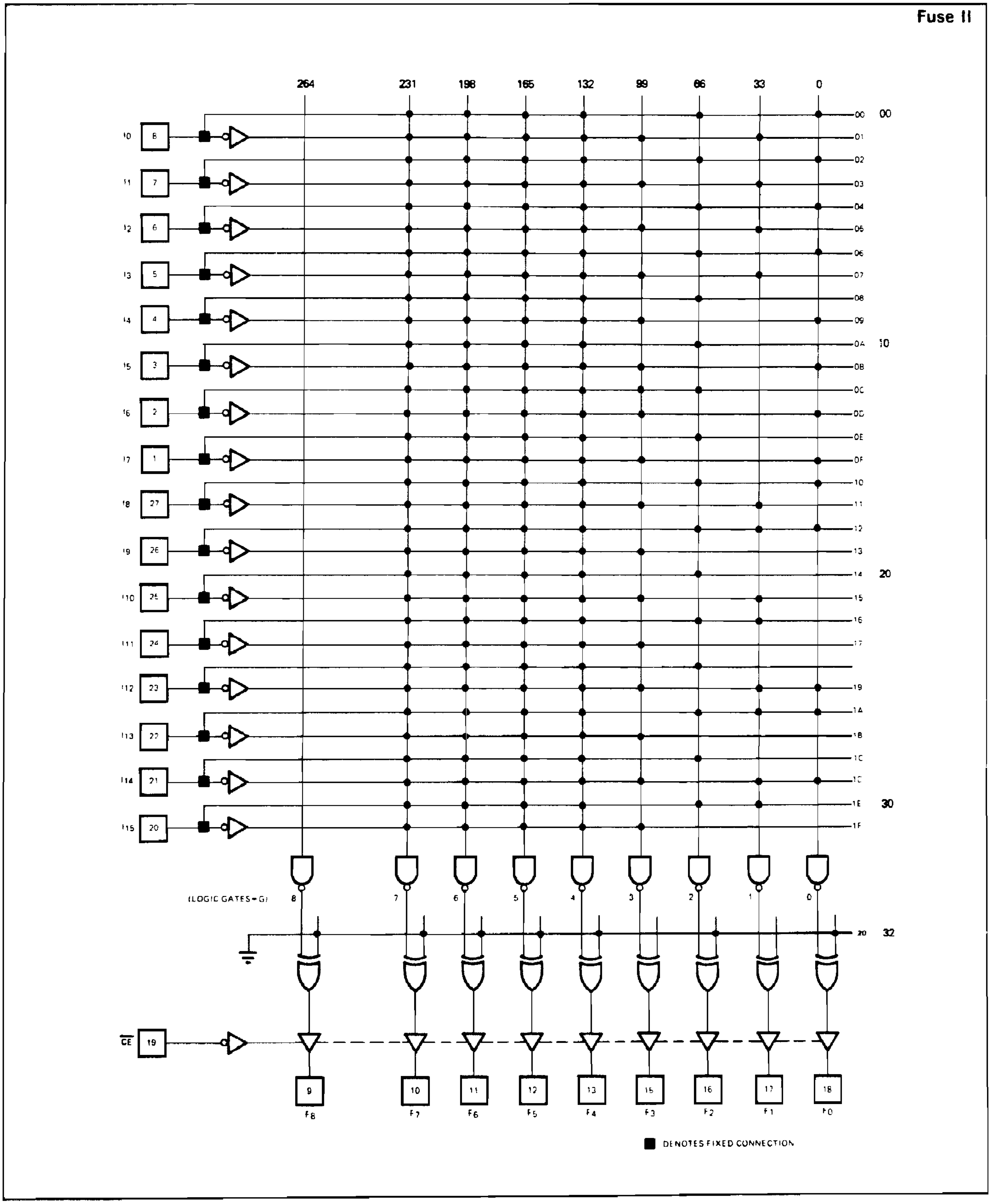
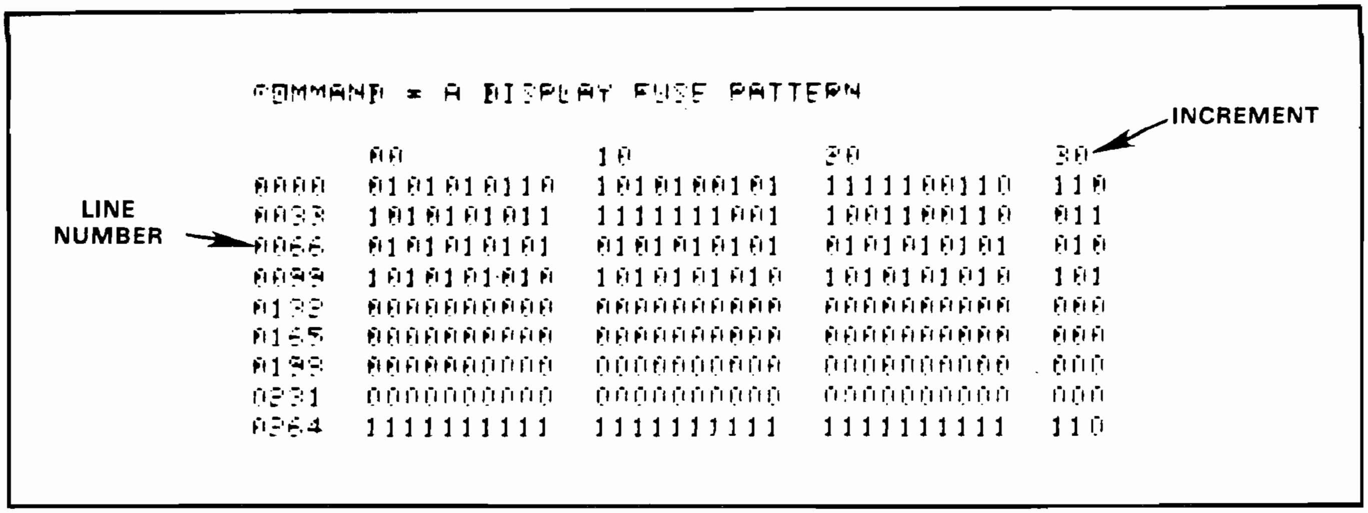


Figure 3-13. Logic Diagram



```
COMMAND = A DISPLAY FUSE PATTERN

         00          10          20          30
0000   0101010110  1010100101  1111100110  110
0033   1010101011  1111111001  1001100110  011
0066   0101010101  0101010101  0101010101  010
0099   1010101010  1010101010  1010101010  101
0132   0000000000  0000000000  0000000000  000
0165   0000000000  0000000000  0000000000  000
0198   0000000000  0000000000  0000000000  000
0231   0000000000  0000000000  0000000000  000
0264   1111111111  1111111111  1111111111  110
```

Line Number + Increment =
Fuse Number

Fuse State
0 = intact
1 = open

Figure 3-14. Display Fuse Pattern

3.5.5 ENTER FUNCTIONAL TEST DATA (8).

This function lets you view or change the parameters for the *Logic Fingerprint™ test and the* Structured Test (fig 3-15). These two tests and key sequences are described in the LogicPak Manual Section on Testing Select functions.

Documentation Documenting the fuse data in RAM can be accomplished in three ways: (1) Transmit Signetics Translator, (2) Display Fuse Data (already discussed), and (3) Transmit Fuse Data. The key sequence is as follows:

1. Complete all necessary setup procedures (sec. 3.3).

2. Set up the unit that is to receive the data.

3. Enter 3 or C.

4. The programmer will output fuse data in RAM to the serial port.

3.5.6 TRANSMIT SIGNETICS TRANSLATOR (3)

This function downloads the fuse states in the programmer RAM in ASCII-logic (Signetics translator) format. Figure 3-16 shows the output formats for the Signetics series of devices.

3.5.7 TRANSMIT FUSE DATA (C)

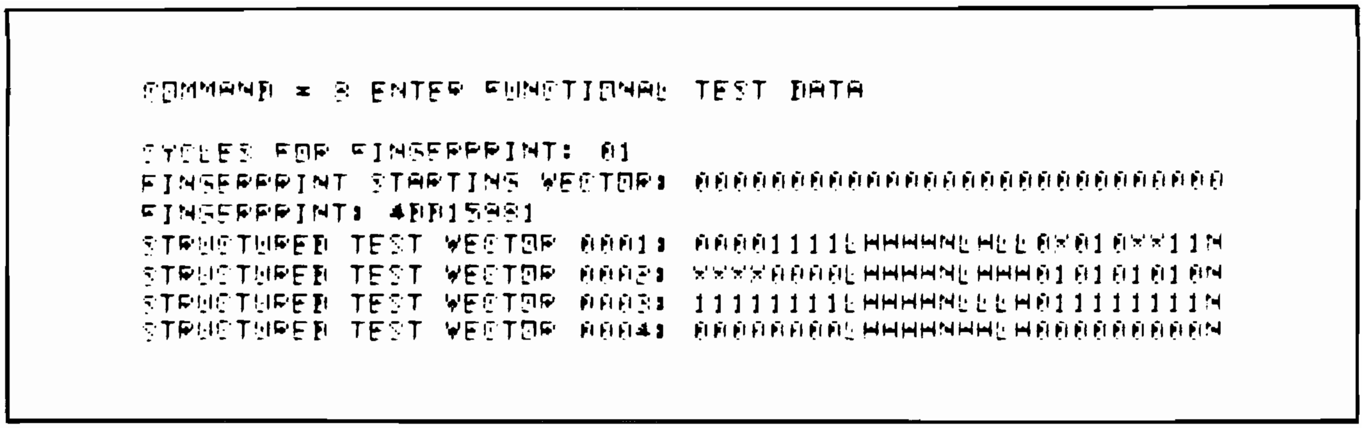
This function outputs the fuse states in the programmer RAM in the JEDEC format. Figure 3-17 shows an example of the fuse data transmission, the JEDEC format is explained in the LogicPak manual.

3.5.8 ENTER REJECT COUNT OPTION

See LogicPak Manual section on Operational Select functions.

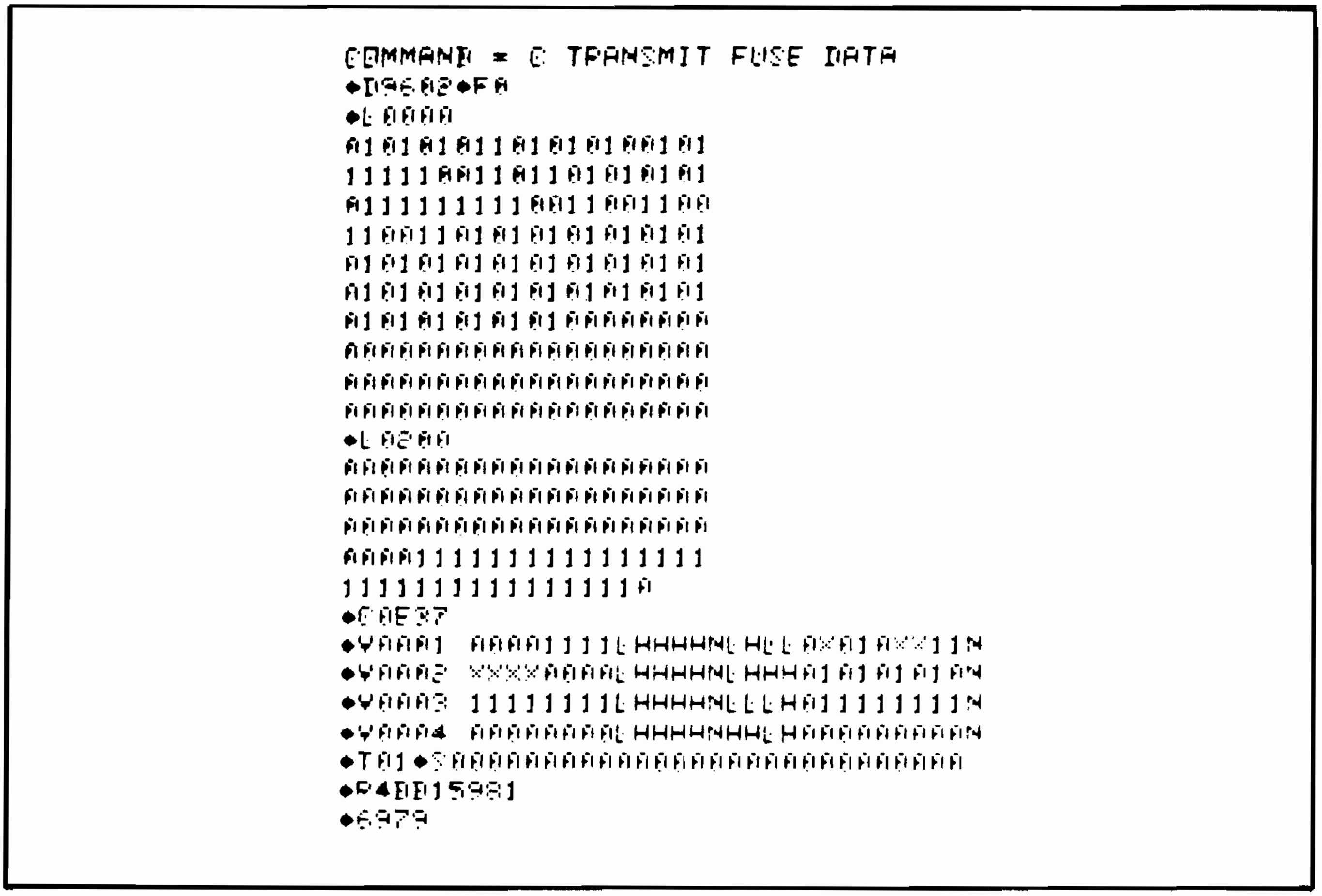
3.5.9 ENTER VERIFY OPTION

Both of the above programming functions are included as a convenience in the design adapter firmware. The options are stored in RAM and cannot be stored as part of a logic format. Consult the LogicPak manual Section on Testing Select Options for additional information.



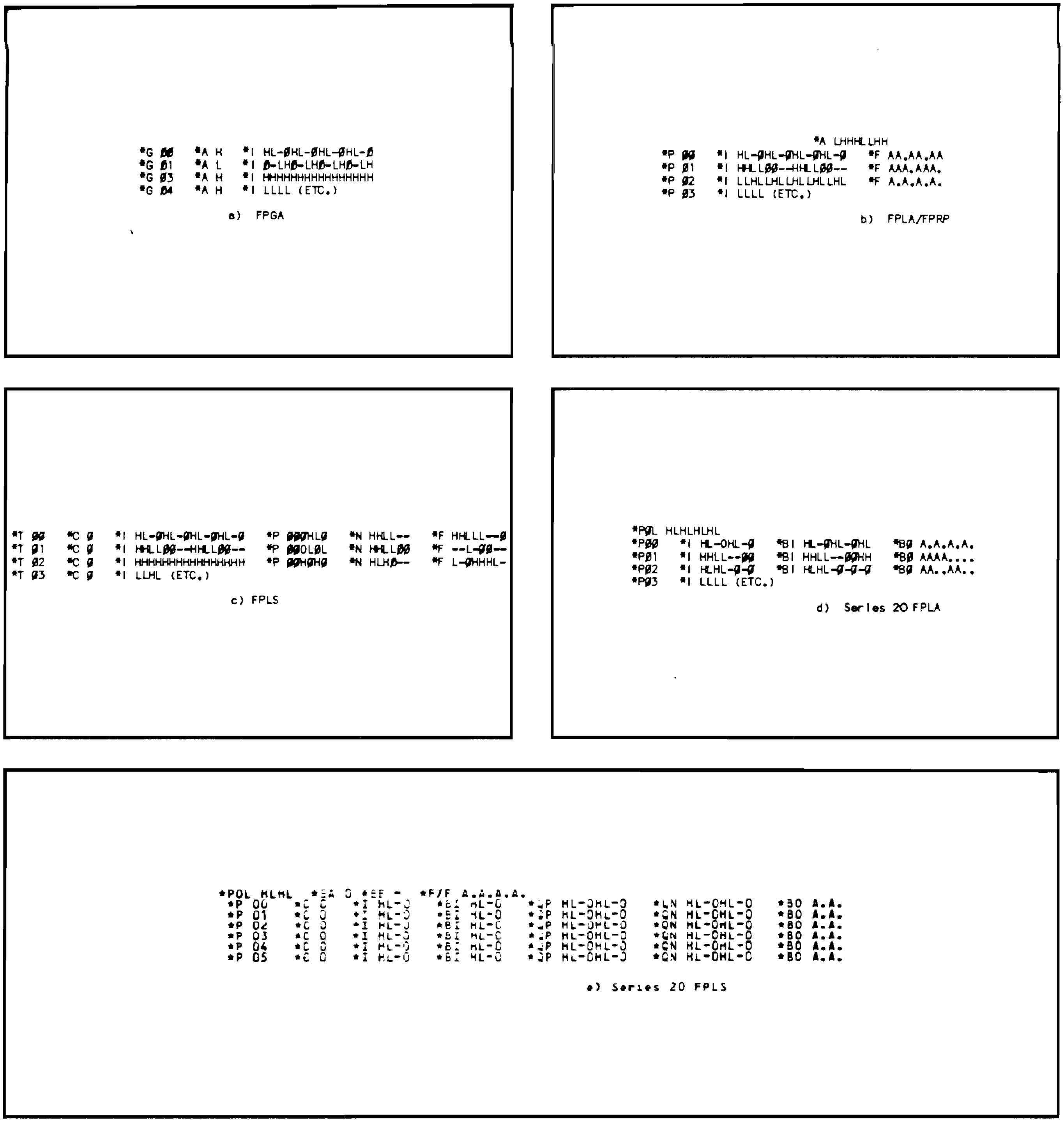
```
COMMAND = 8 ENTER FUNCTIONAL TEST DATA

CYCLES FOR FINGERPRINT: 01
FINGERPRINT STARTING VECTOR: 0000000000000000000000000000
FINGERPRINT: 4DD15981
STRUCTURED TEST VECTOR 0001: 00001111LHHHHNLHLL0X010XX11N
STRUCTURED TEST VECTOR 0002: XXXX0000LHHHHNLHHH01010101 0N
STRUCTURED TEST VECTOR 0003: 11111111LHHHHNLLH0111111111N
STRUCTURED TEST VECTOR 0004: 00000000LHHHHNHHLH000000000N
```

Figure 3-15. Enter Functional Test Data



```
COMMAND = C TRANSMIT FUSE DATA
◆D9602◆F0
◆L0000
01010101101010100101
11111001101101010101
01111111110011001100
11001101010101010101
01010101010101010101
01010101010101010101
01010101010100000000
00000000000000000000
00000000000000000000
00000000000000000000
◆L0200
00000000000000000000
00000000000000000000
00000000000000000000
00001111111111111111
111111111111111110
◆C0E37
◆V0001 00001111LHHHHNLHLL0X010XX11N
◆V0002 XXXX0000LHHHHNLHHH010101010N
◆V0003 11111111LHHHHNLLLH011111111N
◆V0004 00000000LHHHHNHHLH000000000N
◆T01◆S0000000000000000000000000000
◆P4DD15981
◆6979
```

Figure 3-17. Transmit Fuse Data



```
*G ØØ    *A H    *I HL-ØHL-ØHL-ØHL-Ø
*G Ø1    *A L    *I Ø-LHØ-LHØ-LHØ-LH
*G Ø3    *A H    *I HHHHHHHHHHHHHHHH
*G Ø4    *A H    *I LLLL (ETC.)
```

a) FPGA

```
                                 *A LHHHLLHH
*P ØØ    *I HL-ØHL-ØHL-ØHL-Ø    *F AA.AA.AA
*P Ø1    *I HHLLØØ--HHLLØØ--    *F AAA.AAA.
*P Ø2    *I LLHLLHLLHLLHLLHL    *F A.A.A.A.
*P Ø3    *I LLLL (ETC.)
```

b) FPLA/FPRP

```
*T ØØ   *C Ø   *I HL-ØHL-ØHL-ØHL-Ø   *P ØØØHLØ   *N HHLL--   *F HHLLL--Ø
*T Ø1   *C Ø   *I HHLLØØ--HHLLØØ--   *P ØØOLØL   *N HHLLØØ   *F --L-ØØ--
*T Ø2   *C Ø   *I HHHHHHHHHHHHHHHH   *P ØØHØHØ   *N HLHØ--   *F L-ØHHHL-
*T Ø3   *C Ø   *I LLHL (ETC.)
```

c) FPLS

```
*PØL HLHLHLHL
*PØØ    *I HL-OHL-Ø    *BI HL-ØHL-ØHL    *BØ A.A.A.A.
*PØ1    *I HHLL--ØØ    *BI HHLL--ØØHH    *BØ AAAA....
*PØ2    *I HLHL-Ø-Ø    *BI HLHL-Ø-Ø-Ø    *BØ AA..AA..
*PØ3    *I LLLL (ETC.)
```

d) Serles 20 FPLA

```
*POL HLHL  *EA 0 *EP -  *F/F A.A.A.A.
 *P 00    *C 0    *I HL-0   *BI HL-0   *QP HL-0HL-0   *QN HL-0HL-0   *BO A.A.
 *P 01    *C 0    *I HL-0   *BI HL-0   *QP HL-0HL-0   *QN HL-0HL-0   *BO A.A.
 *P 02    *C 0    *I HL-0   *BI HL-0   *QP HL-0HL-0   *QN HL-0HL-0   *BO A.A.
 *P 03    *C 0    *I HL-0   *BI HL-0   *QP HL-0HL-0   *QN HL-0HL-0   *BO A.A.
 *P 04    *C 0    *I HL-0   *BI HL-0   *QP HL-0HL-0   *QN HL-0HL-0   *BO A.A.
 *P 05    *C 0    *I HL-0   *BI HL-0   *QP HL-0HL-0   *QN HL-0HL-0   *BO A.A.
```

e) Series 20 FPLS

Figure 3-16. Output Formats