

# Optima

Universal Programming System

---

---

User Manual



096-0222-002

# Optima

---

Universal Programming System

**User Manual**



096-0222-002

March 1999

096-0222-002

Data I/O has made every attempt to ensure that the information in this document is accurate and complete. Data I/O assumes no liability for errors, or for any incidental, consequential, indirect, or special damages, including, without limitation, loss of use, loss or alteration of data, delays, or lost profits or savings, arising from the use of this document or the product which it accompanies.

No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without written permission from Data I/O.

Data I/O Corporation  
10525 Willows Road N.E., P.O. Box 97046  
Redmond, Washington 98073-9746 USA  
425-881-6444

**[www.dataio.com](http://www.dataio.com)**

Acknowledgments:

Data I/O is a registered trademark and TaskLink is a trademark of Data I/O Corporation.

Data I/O Corporation acknowledges the trademarks of other organizations for their respective products or services mentioned in this document.

©1999 Data I/O Corporation  
All rights reserved

# Table of Contents

## 1. Introduction

Overview .....	1-1
Standard Features.....	1-2
System Components .....	1-2
PC Requirements .....	1-2
Conventions Used in this Manual.....	1-3
A Quick-Start For Optima for Windows.....	1-3
Setting Up the Optima Programmer .....	1-3
Installing Optima Windows.....	1-4
Running Optima Windows Self-test & Calibration .....	1-5
Installing the Optima Booster for a Slow Printer Port .....	1-7
Upgrading Optima Windows .....	1-8
When to Use Optima DOS or ETU .....	1-8
Programming a Memory Device (PROM) .....	1-9
Programming a Programmable Logic Device (PLD) .....	1-12
Gang and Set Programming Using the TOP432DIP .....	1-14
A Quick-Start For Optima DOS Software .....	1-18
Default Path and File Extensions .....	1-18
File Directories .....	1-18
File Input Mode.....	1-18
JEDEC Scan Mode.....	1-19
User Interface Notes.....	1-19
Making Custom Interfaces .....	1-19
Self-test.....	1-20

## 2. Getting Started

Hardware Installation .....	2-1
Software Installation .....	2-1
Files in the Sprint Directory .....	2-3
Computer Set-up .....	2-3

## 3. Optima Tutorial

Logic Devices .....	3-2
Selecting a Device.....	3-2
Commands.....	3-3
Memory Devices.....	3-6
Selecting a Device.....	3-6
Commands.....	3-7

**4. Logic (PLD) Functions**

Command Line Options.....	4-1
Device Selection .....	4-2
Built-In 'Wall Chart'.....	4-3
Main Programming Menu .....	4-4
Display Error Map .....	4-4
PLD Programming Commands.....	4-5
PLD Error Messages.....	4-18
Troubleshooting Test Vector Errors .....	4-19
JEDEC File Standards .....	4-20
Device Specific Information.....	4-21

**5. Memory Device Functions**

Command Line Options.....	5-1
Device Selection .....	5-2
Automatic Device Selection.....	5-3
Built-In 'Wall Chart'.....	5-4
Main Programming Menu .....	5-5
PROM Programming Commands .....	5-5
Input File Formats.....	5-11
PROM Error Messages .....	5-17
Device Specific Information.....	5-18

**6. Optima Technical Specifications**

Self-test .....	6-2
Socket Pinout.....	6-3

**7. Sprint Error Messages**

Error Responses .....	7-1
PLD Error Messages.....	7-2
PROM Error Messages .....	7-3

# 1. Introduction

## OVERVIEW

Congratulations on purchasing this Data I/O programmer. You will now experience the ease of use and flexibility that has made Data I/O the leading personal programmer for professional users. This manual will guide you through the process of setting up and running any of these engineering and production products in the Optima family:

<b>Singlesyte Optimas</b>	<b>Multisyte Optimas</b>
<ul style="list-style-type: none"><li>• Plus 48</li><li>• Optima</li><li>• Optima Light</li></ul>	<ul style="list-style-type: none"><li>• Dual</li><li>• Quad</li><li>• Octal</li></ul>






All the programmers in this family, except the Plus 48, will accept any one of the interchangeable TOP socket modules. They can also be controlled, whether Singlesyte or Multisyte, with the same simple, intuitive screen controls of the following software:

- Optima Windows<sup>®</sup>
- Optima DOS
- Easy-To-Use (ETU)—for production environments

To get started using this universal hardware and software you simply connect one of the programmers to your PC, and then leave it to the software to identify the programmer. Your Windows dialog box will graphically display which TOPs and which programmer from the Optima family you are currently using.

**Note:** *Most of the software commands described in this manual apply to all the programmers in the Optima family. If a command or operation only applies to one of the programmers, we will label it to let you know. If there is no special note, then you can use the command in the same way regardless of whether you have a Plus48 or an Octal connected on the other end of your PC cable.*

As a peripheral to your PC, the programmer utilizes the RAM, CPU, and disk drives in the computer for user interface and data exchange. It comes with its own power supply and timing circuits for accurate generation of programming signals. Any IBM compatible DOS computer can control the programmer.

Included with the programmer are the Optima Windows<sup>®</sup> and the Optima DOS software programs. Begin operation by using Optima Windows as instructed below. Optima Windows runs programmer tests, and programs your devices. Certain special functions require optional Optima DOS or ETU software. For more information, refer to "When to Use Optima DOS or ETU" in this chapter.

**STANDARD FEATURES:**

- High Pin Count. Standard 48 pin (300/600 mil) DIP socket supports devices from 8-48 pins.
- Universal ASIC Pin Drivers. These provide precise digital and analog signals on every pin. Controlled rise times and advanced noise suppression circuitry assure high quality, reliable programming.
- Quick Response. Single keystroke operation and instantaneous file load times make Optima both quick to learn and easy to use.
- Device Support. PROMs, EPROMs, EEPROMs, PLDs, EPLDs, EEPLDs, GALs, FPLAs, IFLs, MCUs, Flash and more in 3 and 5 volt technologies, and in various package types.
- PC peripheral. Optima uses the computer's existing RAM, Disk, Keyboard and Display to provide maximum price/performance.
- Relay Power/Ground. Optimal noise-free device power for highest yields.
- Fast Clocks. Fast rise times (3 ns) allow reliable vector testing.
- State Machine Waveform Generation. Internal crystal controlled state-machine insures accurate programming waveforms not dependent on PC timing.
- Full Screen Menus make the OPTIMA easy to learn and operate.
- Customize User Interface Option. All input commands can be executed from a batch file, or via a network or multitasking Windows environment, on a local or remote PC.

**SYSTEM COMPONENTS**

Before proceeding to installation, please verify that your package contains the following.

- The programmer
- Optional TOPs (Quad and Octal programmers only)
- 25 pin, 1:1 parallel cable
- Optima-Booster (needed only if a "slow printer" message appears during Self-test)
- External power supply (Quad and Octal programmer only) and power cord
- Optima Windows<sup>®</sup> and Optima DOS on CD
- Optima User manual

**PC REQUIREMENTS**

DOS-based PC, AT, 486, Laptop, or Notebook.  
640K RAM (minimum)  
Hard disk with 10 Megabyte of space available.  
VGA or LCD.

## CONVENTIONS USED IN THIS MANUAL

This manual describes both Optima Windows and Optima DOS. Throughout this document, Windows elements and messages will be in **Bold** type. For example, "Click **SMP-Update**. The **Enter update code number** box appears."

DOS commands will be in **BOLD ITALIC type** followed by the file name, etc. in CAPS then **<Enter>**. For example, *DIR \*.CNF* **<Enter>** would list all of the configuration files on your screen.

Optima DOS messages will be in **BOLD** type. For example, **VERIFY DEVICE**. Inputs expected from the user will be **BOLD** type and displayed in brackets **<xx>**. For example **<n>** indicates an Optima DOS input command. Optima DOS accepts both upper and lowercase input commands.

Optima DOS displays prompts if the user is requested to answer a question. Optima DOS shows a Capital letter for the default response, and a small letter for the optional responses. For example, **[Y/n/#]** indicates that the yes will be the default, but you can also enter **n** or **#**. The user will be prompted on screen for the meanings of the different options.

## A Quick-Start For Optima Windows®

The next several sections describe how you set up and operate the programmer and Optima Windows® software. We will review the basics you need to understand so you can set up the programmer and start programming devices quickly and easily.

### 1. Setting Up the Programmer

Follow these steps to set up the programmer.

- 1) Connect the programmer to your PC using the enclosed cable. Insert one end of the cable into the connector on the back of the programmer (either end is okay). Insert the other end into the parallel port of your computer. If your computer has 2 printer ports, either can be used.

Some parallel ports may have hardware characteristics that cause a problem communicating with the programmer. If you see the message, "found slow printer port." during installation or the self-test, install the Optima-Booster supplied with your programming. Follow the instructions, "Installing the Optima-Booster for a Slow Printer Port" in this chapter.

- 2) If you are using an Optima, Optima Light, Plus 48, or Dual programmer, connect the small DIN jack of the External Power supply into the power connector on the back of the programmer. This jack is keyed and cannot be inserted incorrectly. The Quad and Octal programmers have internal power supplies and do not require this step.
- 3) Connect the programmer to a 100 to 250 Volts, 50-60 Hz AC outlet using the enclosed power cord.

**Note:** *Do not turn on the power yet.*

- 4) Programmers come with a variety of TOPs. If the TOPs that came with your programmer are not already installed, install them now onto the programmer. Make sure that the TOPs support the package type of the devices that you will be programming.

**Note:** *You can arrange the TOPs in any order you want. It is common to arrange similar TOPs close to each other.*

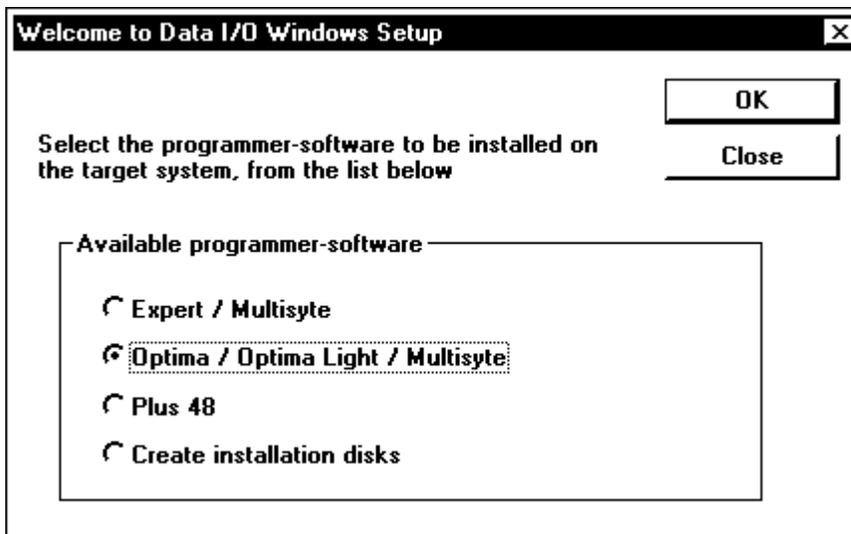
- 5) Turn the power switch that is on the back of the programmer to the ON position.

## 2. Installing Optima Windows

The Optima Release CD contains different installation programs for the Windows and DOS versions: SetupWin.exe and SetupDOS.exe. You can also create installation disks for a computer that does not have a CD-ROM drive, and run Setup.exe. Follow these steps to install Optima Windows on your PC.

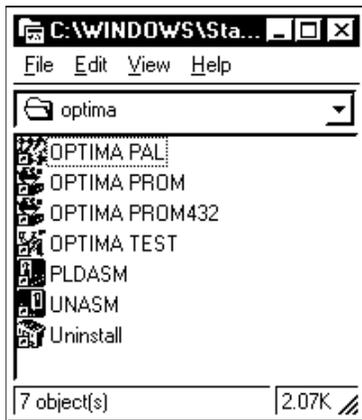
- 1) Insert the Optima Release CD, or installation Disk 1 that you created, into the appropriate drive on your PC. If needed, use a PC that has a CD-ROM drive to create installation disks by following these instructions.
- 2) If you inserted installation Disk 1, click the **Start** button, click **Run**, enter **<drive>:lsetup**, where **<drive>** is the disk drive holding Disk 1 (for example, type **a:lsetup**), then click **OK**. This displays the installation program.
- 3) If you inserted the CD, Windows starts the SetupWin.exe installation program and you will see the **Welcome to Data I/O Windows Setup** dialog box (displayed below). Select one of the following options, then click **OK**:
  - **Optima / Optima-Light / Multisyte**—most cases
  - **Plus 48**—if connected to a programmer marked PLUS48
  - **Create installation disks**—if installing onto a PC that does not have a CD-ROM drive

*Note: Do not select the **Expert/Multisyte** option. It was for older hardware configurations that are now unsupported.*



- 4) During installation the system asks you to name an installation directory. We recommend that you choose the default directory, **C:\Optima**. The **Set-up Type** dialog box is displayed.
- 5) Select **Typical**.

- 6) The system creates the seven icons shown below.



- 7) Drag the first four icons (**Optima Prom**, **Optima Pal**, **Optima Test**, and **Optima Prom432**) to the Windows desktop.

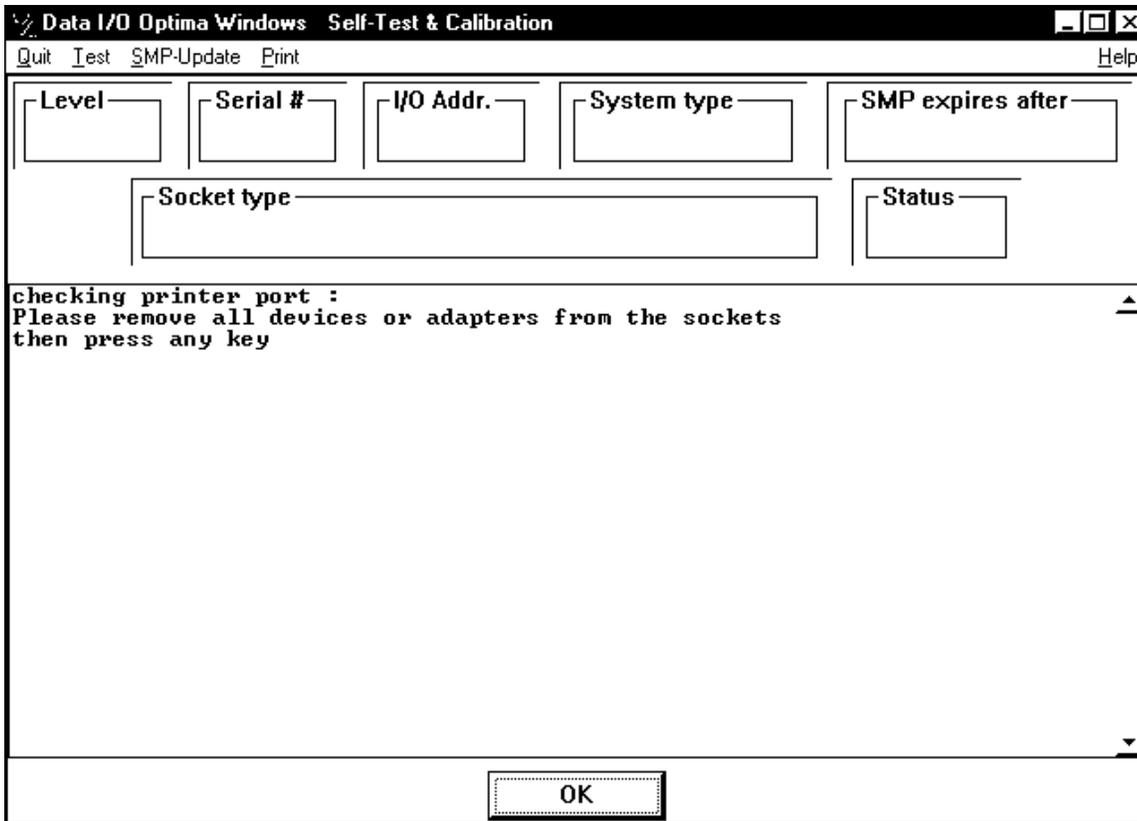
**Note:** *PldAsm* and *UnAsm* are older utilities that are no longer supported by Data I/O. Do not move these to the Windows desktop.

- 8) Close all Optima applications.

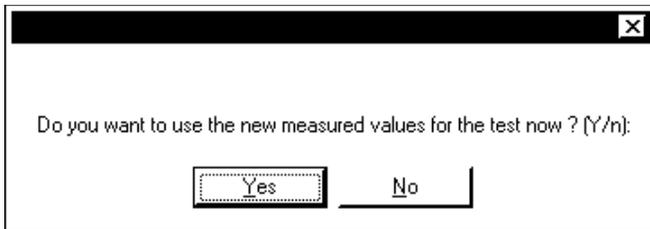
### 3. Running Optima Windows Self-test & Calibration

After Optima Windows is installed you will need to initialize it so it can establish communications with the programmer and load the basic system configuration.

- 1) Make sure that all Optima applications have been closed. Otherwise you will be unable to continue.
- 2) Double-click the **Optima Test** icon to start the self-test. The **Self-Test & Calibration** dialog box is displayed (displayed below). If you see a message that Windows can start only one copy, press **Ctrl+Alt+Del**, select the **Otestw** task in the list, then click **End Task**. It may take a few seconds for Windows to ask you to confirm this command for the task.



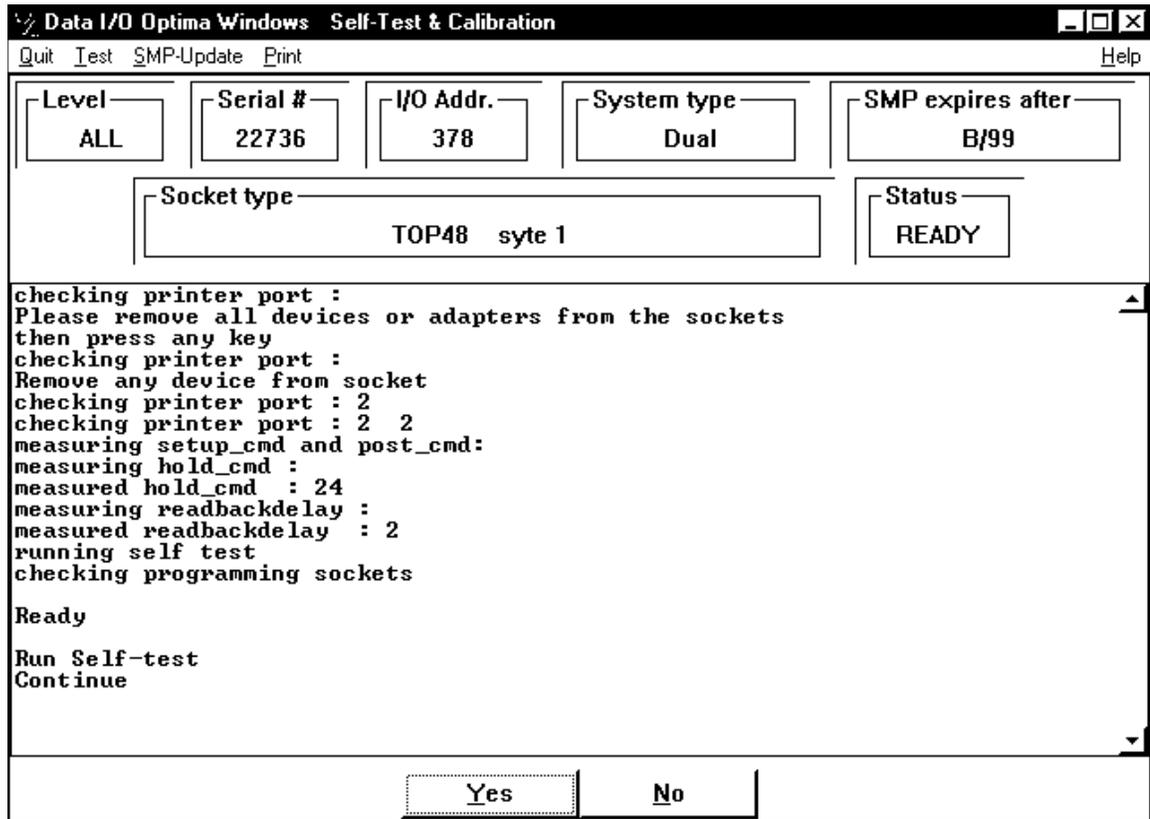
3) Follow the instructions on the window, or click **OK** or **Yes**. If the following window appears, click **Yes**.



**Note:** If the test software unexpectedly closes, make sure that the programmer is connected to your computer, end the Otestw task, then restart Optima Test.

**Note:** If you see the message, **found slow printer port**, anytime during installation or during the self-test, install the Optima-Booster according to the instructions below.

- 4) When the self-test has initialized the system, information such as the programmer's serial number, the system type, and maintenance expiration date is displayed.



- 5) Click **Yes** to complete all the system tests. The system configuration and the results of the tests are displayed on the screen when they are completed.
- 6) Close the test software if you are prompted to do so.

## Installing the Optima Booster for a Slow Printer Port

You may have seen the message, **found slow printer port**, during installation or during the self-test. This means that the parallel port is having a problem communicating with the programmer. As a solution, your system includes an extra piece of hardware called the Optima-Booster, which has extra drive features and filters that minimize hardware interference. Follow these steps to install the Optima-Booster onto you PC.

- 1) Turn off the programmer and connect the end of the booster labeled PC-SIDE directly to one parallel port on the PC.

**CAUTION:** Do not install the booster onto a COM, SCSI, or multiple parallel (LPT) ports. Otherwise damage to the booster or PC might result.

**Note:** Make sure that the booster is connected directly to the PC, not the programmer.

- 2) Connect the cable into the other end of the booster, then turn the programmer back on.

**Note:** Some laptops do not have enough power to have a booster installed.

## Upgrading Optima Windows

To upgrade your Optima software, follow the instructions “Installing Optima Windows” at the beginning of this chapter. Before you do, you may want to back up the contents of your Optima folder to prevent the new installation from overwriting the previous version. If the upgrade contains new devices or features you want to use, you may need to obtain a key code from Data I/O Corporation before you can use them. Data I/O will provide you one key code per year. Do the following steps to receive and use this code.

To determine whether you need a key code, do the following:

- 1) Run the Optima self-test and note the number in the **SMP expires after** box.
- 2) Click **Help > About** to view the software version number. If the version number is larger than the SMP number, request a key code. Include the serial number in your request.

To enter the code after you have received it, do the following:

- 1) Run the Optima self-test.
- 2) Click **SMP-Update**. The **Enter update code number** box appears.
- 3) Enter the code, then click the **OK** button that is next to the code box.
- 4) Click **OK**, then quit the test software.

## When to Use Optima DOS or ETU

After you have run the self-test and your parallel port is communicating well with the programmer, you may want to begin using the optional Optima DOS or ETU software. First you should decide if you want to repeatedly run the same *jobs* day after day, such as for high-volume production. If so, you should use ETU for creating repeatable jobs. Otherwise continue using Optima Windows or Optima DOS. Your decision to use Optima DOS should be based on the table below.

The term, *jobs*, above refers to a user-defined set of parameters for configuring the programming system to process a set of devices. Jobs include all the options that stay the same and are used each time the devices are programmed. The parameters are saved as a file so that they can be reused even after you quit and restart Optima Windows.

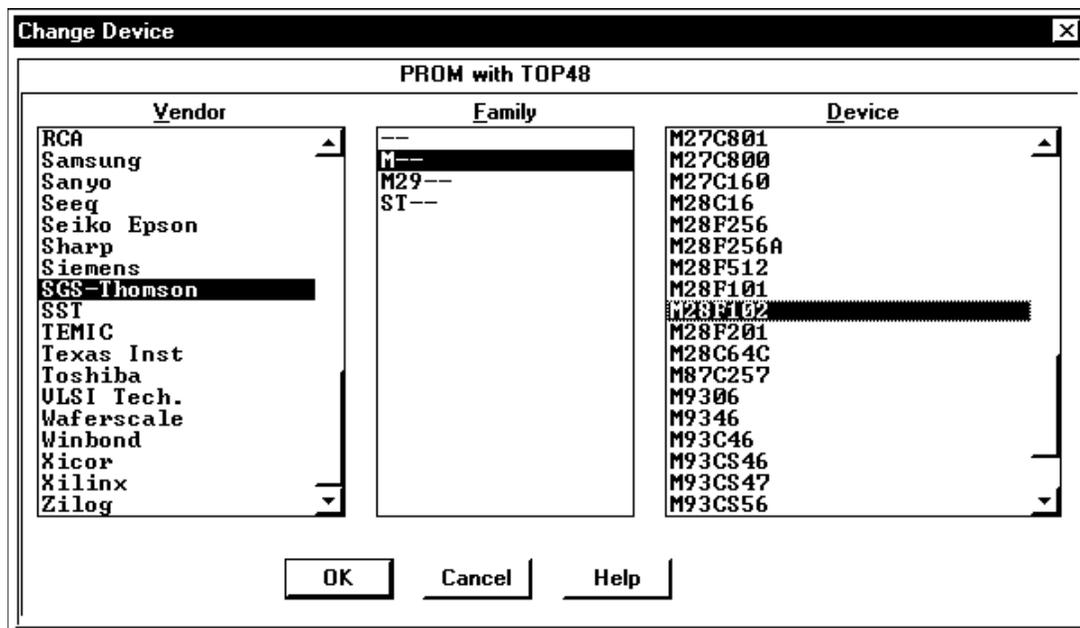
If you want to do this	use this
Save and reuse jobs after quitting and restarting the software	ETU. Refer to the <i>Easy-To-Use Getting Started Guide</i> .
Do not save and reuse jobs, but do any of the following:	
<ul style="list-style-type: none"> <li>• Serialize devices</li> </ul>	Optima DOS. Refer to chapter 2, “Getting Started.”
<ul style="list-style-type: none"> <li>• Display checksums of 5 through 8 characters</li> </ul>	Optima DOS.
<ul style="list-style-type: none"> <li>• Set sector protection bits</li> </ul>	Optima DOS.
<ul style="list-style-type: none"> <li>• Set other device option bits (for example, security bit options for a microcontroller)</li> </ul>	Optima DOS.

**Note:** In Optima DOS, you may see references to the “Plus 48” programmer. Those screens operate Optima family programmers in exactly the same way.

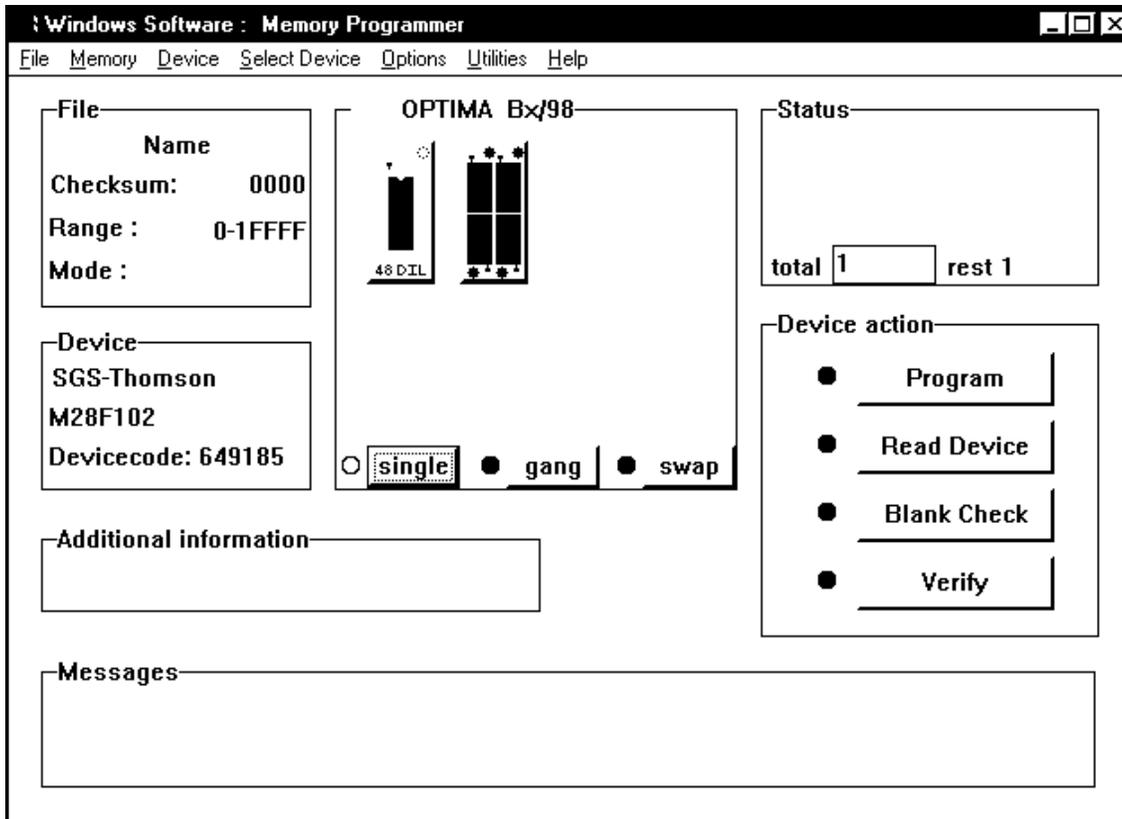
## Programming a Memory Device (PROM)

- 1) Select a device by double-clicking the **Optima PROM** icon on the Windows desktop. A device list for this TOP is displayed.
- 2) To select a device, first choose a **Vendor**. The **Family** and **Device** lists change depending on the vendor you select.
- 3) Choose a general **Family** of devices. The **Device** list may change again based of the family you chose.
- 4) Click on a device part number from the **Device** list. In the screen example below, an SGS-Thomson M28F102 is selected.

**Note:** *Optima Windows detects which devices that the current TOP does not support, and places the word, **TOP!** next to it. If the device you want is not supported, close the software, change the TOP, restart the software, then select the device.*



- 5) When you have selected a device, click **OK** to save that selection and display the main Optima screen (displayed below).

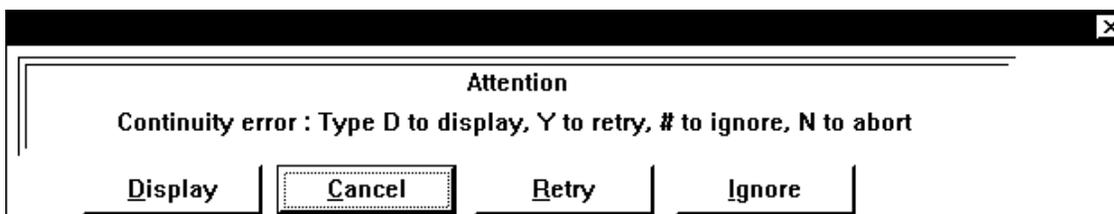


**Note:** Certain package types are not directly supported by a standard Data I/O TOP and can only be programmed if you install an adapter into the TOP socket. The system will prompt you with the name of the required adapter in the **Messages** area.

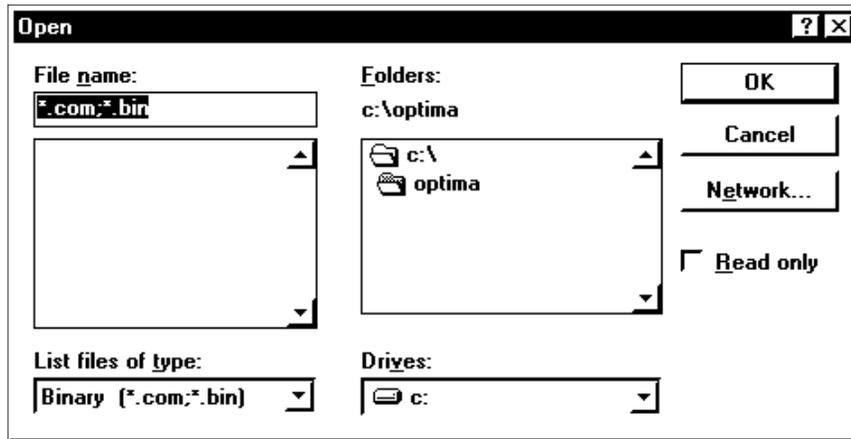
- 6) Before you can program a memory device you will need to fill the programmer's RAM with valid programming data. You can fill RAM by either reading a master device or by loading a data file from one of the PC's drives. If you are loading data from a master device, continue with the next step. If you are loading data from a PC data file, skip the next two steps and continue with step 9 below.
- 7) To read programming data into RAM, insert a master device into the socket that has a green LED next to it turned on.
- 8) Click **Read Device**. Data from the master device is loaded into the programmer's RAM. When the load operation is complete, the **Status** area in the **Memory Programmer** dialog box will display the word **PASS** and the checksum for the data loaded into RAM is displayed.

If you see the **Attention** message, correct any of the following, then click **Retry**.

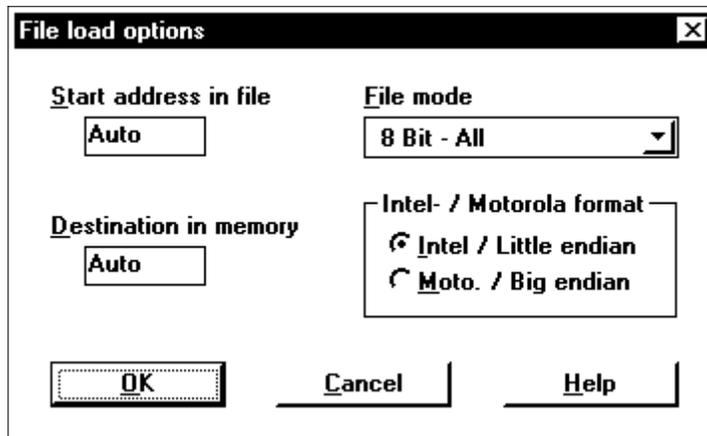
- Device selected in Optima Windows not matching device in the socket
- Device's position in the socket
- Socket lever not in closed position
- Device damage



- 9) If you are loading a PC data file into RAM, select **File > Load** from the main menu bar. The standard Windows **Open** dialog box is displayed.



- 10) Browse through the hard drive and select the appropriate program data file name from the list on the screen. Optima Windows supports many formats, such as Intel HEX and Motorola S Record. If you don't see file name, the file extension is different than those in the **File name** box. Either select a different file type from the **List files of type** box, or type \*.\* in the **File name** box, then click **OK**.
- 11) When you have selected the file, click **OK**. The **File load options** dialog box is displayed.

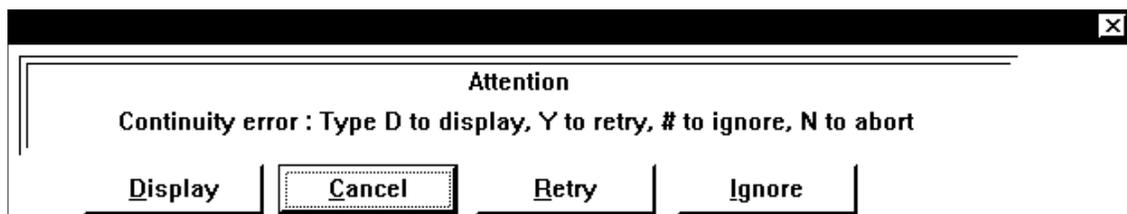


- 12) Leave the default options and click **OK** to return to the main Optima screen.
- 13) Once the data file has been loaded insert a blank device into the socket and click **Program**.

**Note:** Do not use any of the **Algorithm Options** listed under the **Options** menu. These options will change the manufacturer's specifications for programming the device and are used by Data I/O for programming algorithm development.

**Data I/O will not assume any liability or responsibility for devices damaged as a result of being programmed using these Algorithm Options.**

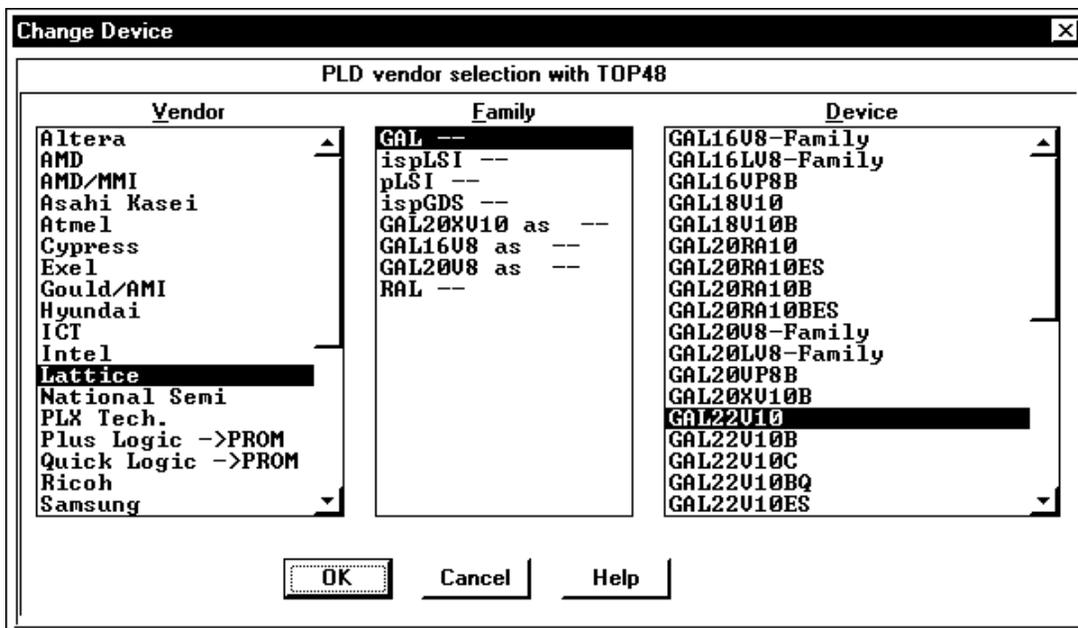
If you see the **Attention** message, correct the problem, then click **Retry**.



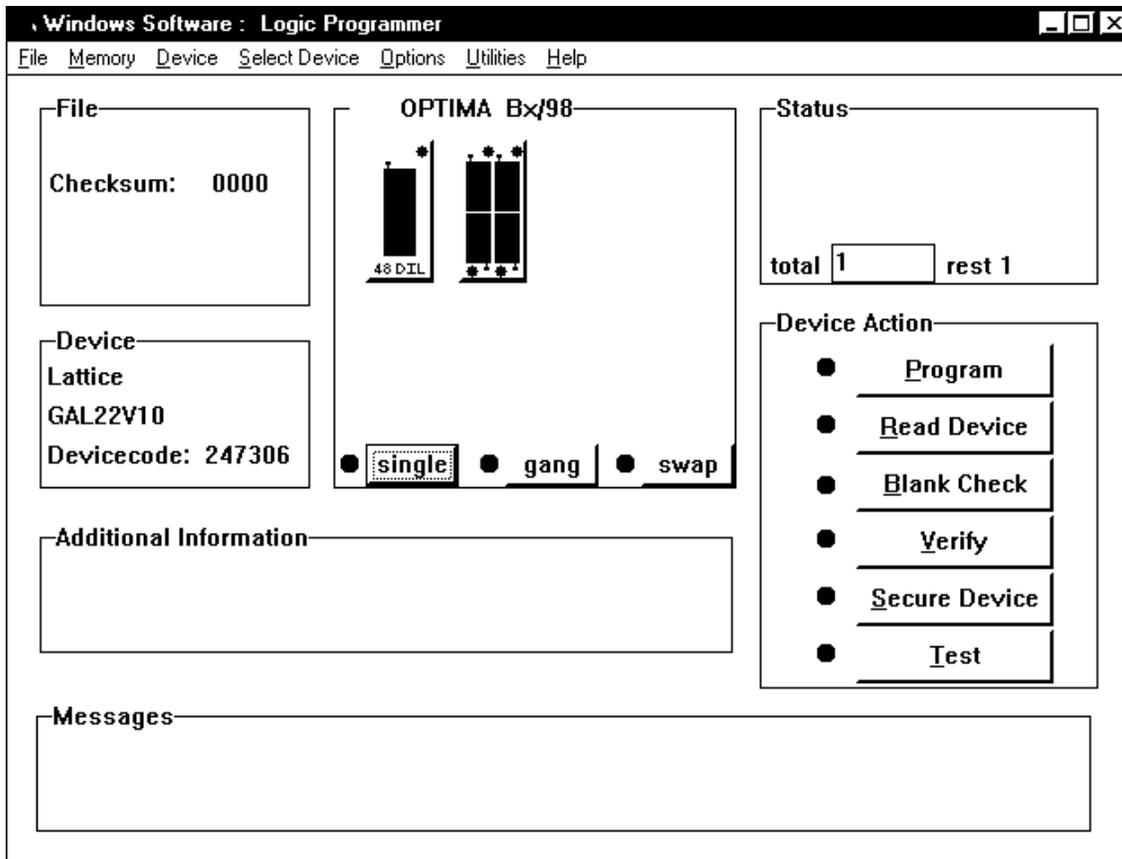
## Programming a Programmable Logic Device (PLD)

- 1) Select a device by double-clicking the **Optima PAL** icon on the Windows desktop. A device list for this TOP is displayed.
- 2) To select a device, first choose a **Vendor**. The **Family** and **Device** lists change depending on the vendor you select.
- 3) Choose a general **Family** of devices. The **Device** list may change again based on the family you chose.
- 4) Click on a device part number from the **Device** list. In the screen example below, a Lattice GAL22V10 is selected.

**Note:** Optima Windows detects which devices that the current TOP does not support, and places the word, **TOP!** next to it. If the device you want is not supported, close the software, change the TOP, restart the software, then select the device.



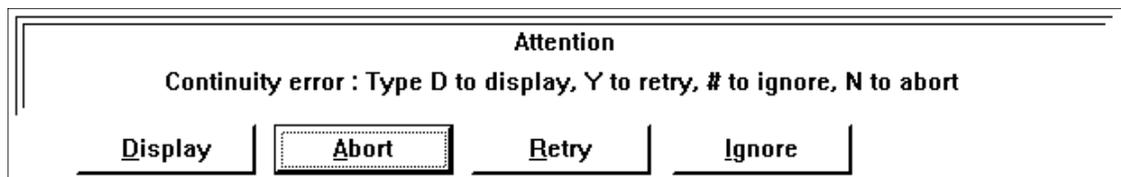
- 5) When you have selected a device, click **OK** to save that selection and display the main Optima screen (displayed below).



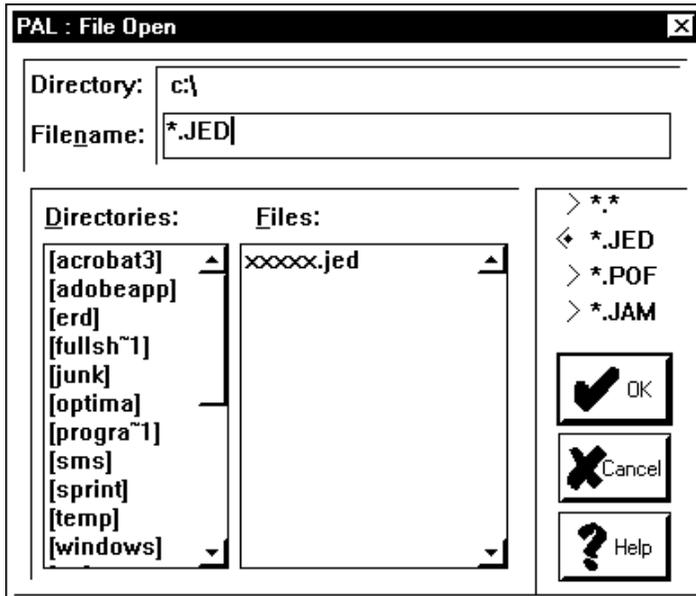
**Note:** Certain package types are not directly supported by a standard Data I/O TOP and can only be programmed if you install an adapter into the TOP socket. The system will prompt you with the name of the required adapter in the **Messages** area.

- 6) Before you can program a PLD device you will need to fill the programmer's RAM with valid programming data. You can fill RAM by either reading a master device or by loading a data file from one of the PC's drives. If you are loading data from a master device, continue with the next step. If you are loading data from a PC data file, skip the next two steps and continue with step 9 below.
- 7) To READ programming data into RAM, insert a master device into the socket that has a green LED next to it turned on.
- 8) Click **Read Device**. Data from the master device is loaded into the programmer's RAM. When the load operation is complete, the **Status** area in the **Memory Programmer** dialog box will display the word **PASS** and the checksum for the data loaded into RAM is displayed.

If you see the **Attention** message, make sure that the correct device is correctly positioned and locked in the upper-left socket, then click **Read Device** again.



- 9) If you are loading a PC data file into RAM, select **File > Load** from the main menu bar. The **File Open** dialog box is displayed.



- 10) Browse through the hard drive and select the appropriate program data file name from the list on the screen. If you don't see file name, the file extension is different than those in the **Filename** box. Either select a different file type from the list, or type \*.\* in the **Filename** box, then click **OK**.
- 11) When you have selected the file, click **OK**.

**Note:** You can gang program PLD devices if you install identical TOPs in the programmer.

- 12) Once the data file has been loaded insert a blank device into the socket and click **Program**.  
If you see the **Attention** message, correct the problem, then click **Retry**.

## Gang and Set Programming Using the TOP432DIP (Memory Devices Only)

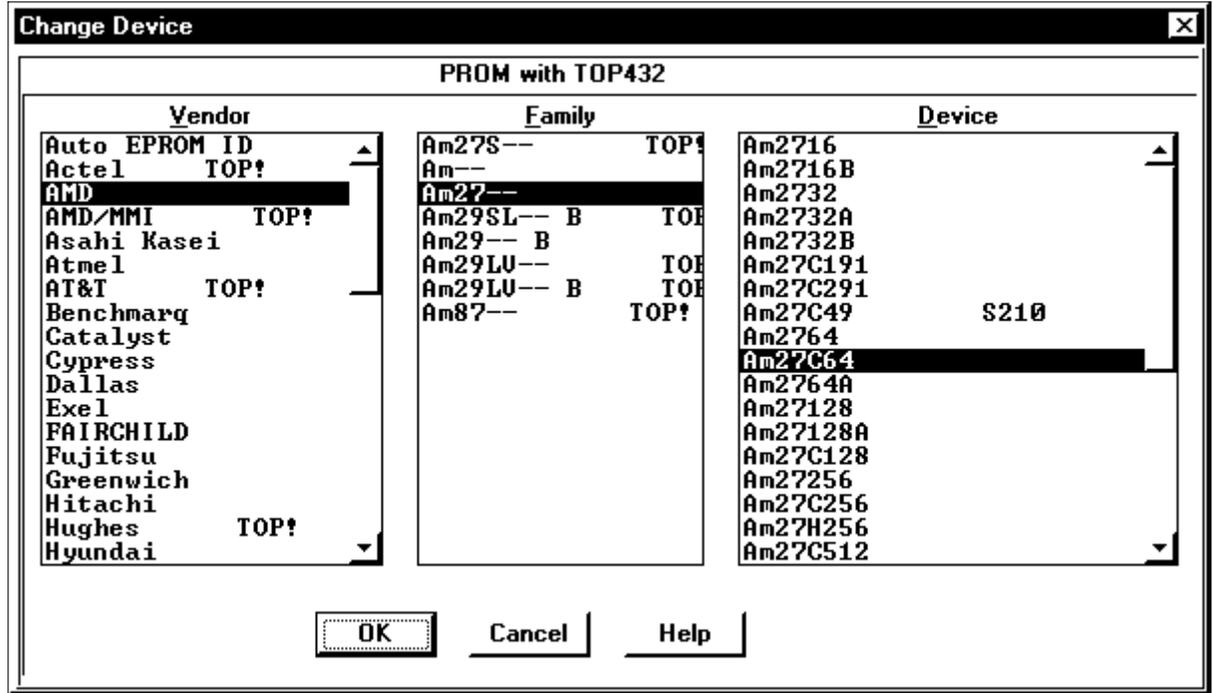
You can program into multiple Flash or EPROM memory devices (up to 32 pins) using either Gang or Set programming.

**Gang programming**—programming the same data into devices.

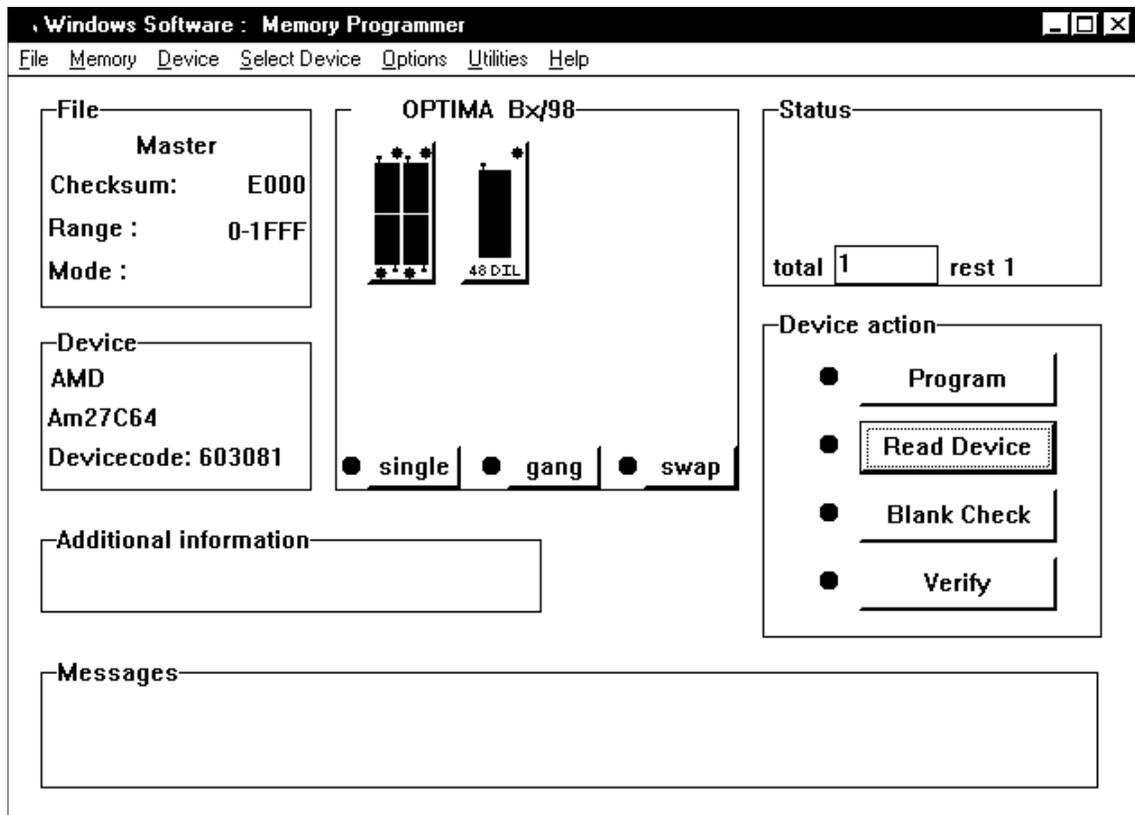
**Set programming**—programming different data into devices.

- 1) Select a device by double-clicking the **Optima PROM432** icon on the Windows desktop when you have TOP432s installed in your programmer. A device list for this TOP is displayed.
- 2) To select a device, first choose a **Vendor**. The **Family** and **Device** lists change depending on the vendor you select.
- 3) Choose a general **Family** of devices.
- 4) Click on a device part number from the **Device** list. In the screen example below, an AMD 27C64 device has been chosen.

**Note:** Optima Windows detects which devices that the current TOP does not support, and places the word, **TOP!** next to it. If the device you want is not supported, close the software, change the TOP, restart the software, then select the device.



- 5) When you have selected a device, click **OK** to save that selection and display the main Optima screen.

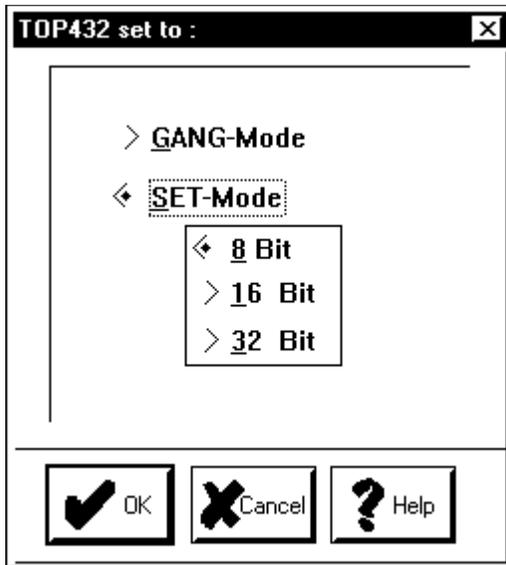


**Note:** Certain package types are not directly supported by a standard Data I/O TOP and can only be programmed if you install an adapter into the TOP socket. The system will prompt you with the name of the required adapter in the **Messages** area.

- 17) If you are doing Set programming, you need to select the correct type. Continue with the next step. If you are Gang programming, skip the next two steps and continue with step 21 below.

**Note:** Set programming can only be done on devices installed in a TOP432. You cannot program a set of four devices with four TOP48 modules (or any TOP except the TOP432) installed in your programmer. However, you can Gang program using any TOP module.

- 18) To select Set programming, click the TOP icon that is highlighted light green. The **TOP432 set to** dialog box is displayed.



- 19) Select the **SET-Mode** option, select one of the following options, then click **OK**:

- **8 Bit**—programs sequential sections of a single program into up to four 8-bit devices.
- **16 Bit**—programs even data into sockets 1 and 3, and odd data into sockets 2 and 4. This is used most often when you have data for a 16-bit application in RAM and you want to program it into two 8-bit devices.
- **32 Bit**—programs byte 0, 4, 8, etc. into socket 1, byte 1, 5, 9, etc. into socket 2, and so on, until 32-bits of data is programmed into four 8-bit devices. Byte 4 is programmed into the second location in the device in socket 1 and so on.

**Note:** When in any of the three SET-modes, the data in memory is arranged in the following way:

byte 0 - socket 1 , byte 1 - socket 2 , byte 2 - socket 3 , byte 3 - socket 4  
 byte 4 - socket 1 , byte 5 - socket 2 , byte 6 - socket 3 , .....

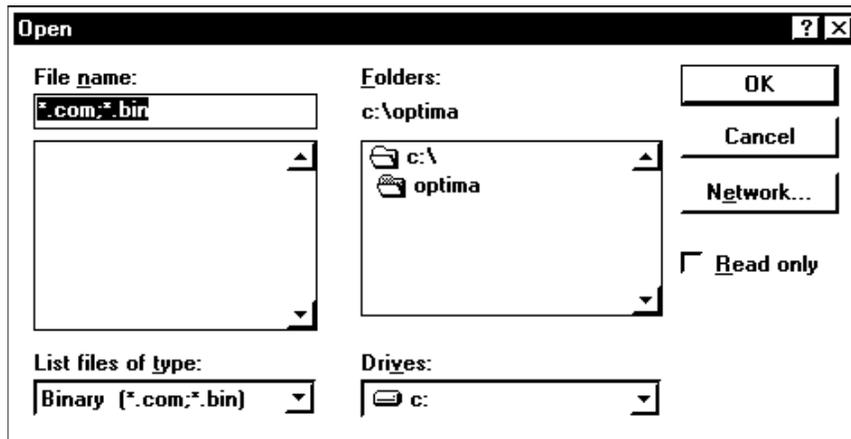
The difference between the different Set modes is only in the file-input. When changing Set modes, the data remains unchanged in memory, therefore always first select **SET-Mode** and then input your data from file.

- 9) Before you can program a PLD device you will need to fill the programmer's RAM with valid programming data. You can fill RAM by either reading master devices or by loading a data file from one of the PC's drives. If you are loading data from a master device, continue with the next step. If you are loading data from a PC data file, skip the next two steps and continue with step 9 below.
- 10) To READ programming data into RAM, insert a master device into the socket labeled **Master**.

- 11) Click **Read Device**. Data from the master devices are loaded into the programmer's RAM. When the load operation is complete, the **Status** area in the **Memory Programmer** dialog box will display the word **PASS** and the checksum for the data loaded into RAM is displayed.

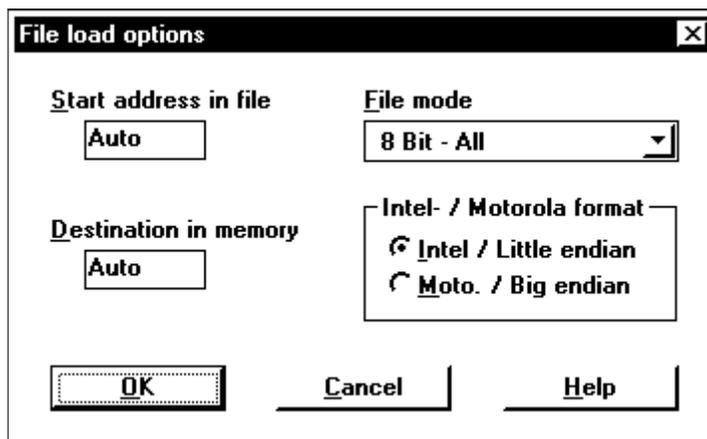
If you see the **Attention** message, make sure that the correct device is correctly positioned and locked in the upper-left socket, then click **Read Device** again.

- 12) If you are loading a PC data file into RAM, select **File > Load** from the main menu bar. The **File Open** dialog box is displayed.



- 13) Browse through the hard drive and select the appropriate program data file name from the list on the screen. Optima Windows supports many formats, such as Intel HEX and Motorola S Record. If you don't see file name, the file extension is different than those in the **Filename** box. Either select a different file type from the list, or type \*.\* in the **Filename** box, then click **OK**.

- 14) When you have selected the file, click **OK**. The **File load options** dialog box is displayed.



- 15) Leave the default options and click **OK** to return to the main Optima screen.
- 16) Once the appropriate data file has been loaded, insert blank devices into all sockets.
- 17) Click **Program**.

If you see the **Attention** message, correct the problem, then click **Retry**.

**Warning:** Do not use any of the **Algorithm Options** listed under the **Options** menu. These options change the manufacturer's specifications for programming the device and are used by Data I/O for programming algorithm development. **Data I/O will not assume any liability or responsibility for devices damaged as a result of being programmed using these Algorithm Options.**

## A Quick-Start for Optima DOS

### DEFAULT PATH AND FILE EXTENSIONS

Optima DOS can be located anywhere in your disk file structure. If you have added Optima DOS to your path, then you can start SPRINT, PAL, PROM etc. from any directory. Optima DOS will look at the directory where it is stored for the configuration files for the board address. If Optima DOS is on a drive with a letter greater than or equal to F, (probably a network drive) then the board address file will be initialized onto the system local drive. The Board I/O address is stored in a file called EXIOADDP.CNF.

### FILE DIRECTORIES

The default path used in Input and Write commands when you first power up Optima DOS is the path from which the program was called. The default extension (file type) depends on the type of device selected. Memory devices default to '.COM' and Logic devices default to '.JED'. These are the typical extensions used for data files that are programmed into these types of devices. The directory information, and other user specified options are stored in either the COM\$\$\$\$.CNF file for PROMs or the JED\$\$\$\$.CNF file for PALs. These files will be stored in the local directory, not in the executable SPRINT file directory.

### FILE INPUT MODE

When a file is to be read, the system will automatically default to the last drive, directory and file type (extension) that was used. All files with the designated extension will be displayed. You may type in the file you want to input or use the arrow keys to select a file or change any of the defaults.

The input screen shows the current drive, current directory, file type option (extension), and those files that match the type selected. Selecting a file will take the user to the next menu. Other entries on the screen with non-matching extensions are directories. The entry '..' refers to a root directory; selecting that entry will step you back the tree towards the root directory, one level at a time. Choosing any other directory entry will select the chosen directory for display.

To input a file you can begin typing the complete path and file name you want to load and then hit the **<enter>** key to initiate loading the file. However, it will be faster to change the defaults to the path where your data files reside and use the cursor to make your selection. Use the arrow keys to move around the screen. If you hit **<enter>** prior to typing in a file, the highlighted entry on the menu will be selected.

Use the TAB key to switch between the drive -, directory -, file type - or the file field.

Setting the cursor onto the drive or directory field allows you to change drive or directory easily. Setting the cursor to the file type field allows you to change the file type, or restrict the type of files shown using standard DOS file names: example: EP\*.JDC.

## JEDEC SCAN MODE

The Optima DOS will detect JEDEC files that are compatible with the PLD you have selected and upon entering the file-input mode, the highlighter will stop on the first file in the directory that is compatible with the device you have selected. If no compatible devices are found, the highlighter will stop at the '..'. If you have a color monitor, we use a color coding to help you move quickly to the right file.

The color-coding used for color monitors are as follows.

PURPLE: Directories  
YELLOW: File is compatible with the device selected  
GRAY: File is not compatible  
GREEN: File is not 100% JEDEC standard but may be compatible with selected device

## USER INTERFACE NOTES

### SINGLE KEYSTROKE RESPONSE

The standard OPTIMA user interface is designed for direct keyboard control. No mouse is required for operation. All menu items have letters, numbers or function keys assigned to them. Activation of any command requires only that the key (for example <P> for program) should be pressed, commands that are non-reversible request confirmation (<Y> to continue). The cursors may be used in file input, editor, device selection, test vector editor and option screens, otherwise simply enter the command you need. We think you will find this to be the fastest approach to operating a tool like the Optima DOS. All input can be in either UPPER or lower case characters.

### SCROLL CONTROL - MORE ?

All commands support controlled scrolling of the screen. When Optima DOS has filled a screen with data, the question **More?** is displayed. Answering <. > or <n> will terminate the command, any other key will allow an additional 20 lines of text to be displayed.

## MAKING CUSTOM USER INTERFACES

### BATCH MODE

Operating Optima DOS from a batch file allows many special, customer defined applications. For example, a serial number could be entered into the memory buffer to allow EPROMs to be serially numbered. A user could embed Optima DOS into a completely different application using the batch mode. In effect, all keystrokes from the keyboard can be replaced by a file on the disk. For our examples we will call this file 'FILEB'. With the batch mode a device can be selected, a file read-in, and a device programmed - all automatically. All OPTIMA programming modules support batch mode. For example, with PROM, batch mode is selected by starting PROM as shown below:

**> PROM -bFILEB**

the execution of 'FILEB' ends either when no more data is in the file, or when a 'Q' character is read while in the main menu. If no Q is found, Batch mode will end without PROM (or PAL) returning to DOS.

## Structure of the SPRINT BATCH File

The first 6 characters of a Batch file are the device codes. A list of device codes can be generated from the Device menu with sequence **<+nyy>**. The use of device codes assures that the batch files need not be changed from one software version to another.

The user can use any HEX editor to create the batch file. The following table shows the special control codes like cursor and function keys.

Function	HEX Value
Cursor Down	0050
RETURN	0D
ESC	1B
F1	003B
F2	003C
F10	0044

The second way to create a BATCH file is with the -R command line option. Simply type:

**PAL -r** or **PROM -r**

at the DOS prompt, and a file called **BATCH** will be created containing all of the keystrokes entered up to and including Q. Avoid using the cursor in this mode in the file input command, it is better to type in the exact directory and file name including extension for storage into the batch file.

EXAMPLES BATCH FILE :

a) blank check Xilinx 1736: (DEVCODE = 661440)

Contents of file FILEB (ASCII) = 661440bq

b) Xilinx 1736 , load file DEMO.COM from C:\SPRINT\EXAMPLES into memory

Contents of the FILEB (ASCII) = 661440ic:\sprint\examples\demo.com  
(Note that there should be a RETURN (0x0D) at the end of these examples)

If any error occurs during batch mode, a 'N' is given as the response any internal error control questions, for example '**skip blank check [N]**' will have a N response in batch mode. When Optima DOS exits to the calling program, there will be a return code with a result code.

## SELF-TEST

### DEVICE INTEGRITY TESTING

Before voltage is applied to a device, there are several tests to determine the integrity of the device inserted in the socket, and the proper operation of the OPTIMA.

- \* Backwards device test will determine if the devices is installed in the socket properly.
- \* Continuity test determines if all of the pins of the device are making proper contact with the pin drivers.
- \* Offset test verifies that the device has the expected number of pins in the package.
- \* Shorts test verifies that no pins are shorted to any other pin in the device.
- \* Pin driver test verifies the operation of the programmer.
- \* Relay contact test.
- \* Device ID testing.

## ERROR RESPONSES

Backward devices or poor socket contact will result in a **pin continuity error message**. The user will be prompted to display the pins that failed. By entering <Y> for yes, the failing pin can be displayed on the screen. The pins in parentheses (**xx**) are the expected ground pins of the device. If the ground contact fails, all pins including (**xx**) will be shown. The reversed device or bad pins can be examined at this point.

***Note:** Not all devices permit continuity testing on all pins. Optima DOS automatically excludes those pins that are not testable. In some cases, the engineer may wish to skip the test and continue programming. This is not recommended, but the special key <#> will allow the user to do so. Normally the operator will type <n> to return to the main menu before correcting the device condition.. Please note that if the error is ignored, damage to the device could occur.*

Device short test will determine if any pins of the installed device are shorted. **Pin XX shorted, continue?** is the error message, always enter <n> since this test provides added protection to the programmer hardware, and a defective device could under certain cases damage the programmer. Please remove and discard the device.

Offset test counts the number of pins in the package, and rejects devices that do have the **WRONG PIN COUNT**. The error condition cannot be skipped.

The programmer checks its own pin drivers before every programming cycle. **VPP PIN XX FAILED** is the message that is displayed. The message is fatal, please run the OPTIMA self-test to determine the exact failure and contact your Data I/O distributor.

Relays are mechanical, and OPTIMA checks the contacts of the relays for stuck open or closed conditions. Should a relay fail, the user will be asked to run the OPTIMA self-test to make a detailed report of the problem.

Assuming all hardware is OK, then the device is check for proper silicon ID. **Incorrect Silicon ID** will result if this happens. There is no option to skip this message. One reason why this error appears is, if you try to program a new version of a device and the Optima DOS is not updated, please contact your local Data I/O distributor for a SPRINT SOFTWARE MAINTENANCE PLAN - SMP.



## 2. Getting Started

### Hardware Installation

- Step 1 Connect the programmer to your PC. For OPTIMA / PLUS48 insert one end of the 25-pin parallel cable into the connector on the back of the programmer (either end is okay). Insert the other end into the parallel port of your computer. If your computer has two printer ports, either can be used.
- Step 2 For OPTIMA / PLUS48 connect the External Power supply. Connect the power supply to a 100 to 250 Volts, 50-60 Hz AC outlet. Connect the cable with the small DIN jack into the power connector on the back of the OPTIMA / PLUS48. This jack is keyed and cannot be inserted incorrectly.

### Software Installation

The Optima Release CD contains different installation programs for the Windows and DOS versions: SetupWin.exe and SetupDOS.exe. You can also copy the DOS installation software, Installh.bat, to disks for a computer that does not have a CD-ROM drive. Optima DOS requires around 10 Megabytes of memory space on your hard disk. Please make sure you have enough space before you begin installing Optima DOS.

- Step 1 Turn on the power to the programmer. The power switch on the back of the OPTIMA / PLUS48 should be in the up position for "on".
- Step 2 Insert the Optima Release CD, or installation Disk 1 that you created, into the appropriate drive on your PC. If needed, use a PC that has a CD-ROM drive to create installation disks by following these instructions. If you see the **Welcome to Data I/O Windows Setup** dialog box, click **Close**.
- Step 3 If you inserted installation Disk 1, run the Installh.bat installation program. For example, at a DOS prompt, enter **a:installh**, then press **<Enter>**.

Step 4 If you inserted the CD, run the SetupDOS.exe installation program. The **Welcome to SOFTWARE Setup** screen is displayed. Select one of the following options, then press **<Enter>**:

- **Optima / Multisyte**—most cases
- **Plus 48**—if connected to a programmer marked PLUS48
- **Copy DOS-software to diskdrive**—if a PC that you want to install onto does not have a CD-ROM drive

**Note:** Do not select the **Expert/Multisyte** option. It was for older hardware configurations that are now unsupported.



Step 5 Follow the prompts through the installation program.

The installation program will create a directory on your hard disk called SPRINT and load all of the files into that directory. Optima DOS will prompt you with an explanation of what it is about to do. If the directory is correct, proceed with the installation by pressing **<Enter>**.

Step 6 Now go back to the hard disk to the directory where you loaded Optima DOS. Type SPRINT **<Enter>**. The opening menu will appear and you will be ready to select any of Optima DOS tools for programming Logic or Memory devices.

Step 7 Start the self-test with **<5>**. Optima DOS will search the address where the programmer is connected. If the correct port address is displayed, enter **<Y>** to accept this address.

OPTIMA / PLUS48 only :

In most cases, the port address displayed during the installation is the correct one. The only time there might be a problem is if you have a second parallel port connected to another peripheral. In this case enter **<n>**.

Step 8 Self-test. The self-test menu now appears. Press **<Enter>** and the programmer will now perform a self-test, checking the hardware and software installation. When complete, the lower red box on the screen should say "PASS".

## FILES IN THE SPRINT DIRECTORY

The following EXE files are installed on your hard disk:

PROM	all memory devices, including Microcontrollers, QuickLogic and Xilinx
PAL	all PLD devices, Bipolar and CMOS
PLDASM	a Boolean to JEDEC file converter
UNASM	a JEDEC file to Boolean equation converter
OPTITEST or TESTEX	Self-test
SPRINT	The main menu
PROM432	Separate memory software for the TOP432 only (not used on the PLUS48)
\Examples	Directory of sample files: JED examples for JEDEC type PALs POF templates for Altera MAX devices HEX examples for memory devices PLD examples for the PLDASM option

**Note:** *PldAsm* and *UnAsm* are older utilities that are no longer supported by Data I/O.

### Computer set-up

Optima DOS is shipped from the factory set-up for a color monitor. If you do not have a color monitor, you may want to change the setting to monochrome. To do this select any Memory device. When you get to the main programming screen, go to the **Display programming values** screen by typing **<O>**. Using the arrow keys, scroll down to the **Screen display mode** and change the option by pressing the space bar until the desired mode is displayed. Once you have the desired mode displayed press **<Enter>** and the change will be recorded in the configuration file. for both PAL and PROM devices. You will not have to change this mode again.

The options are:

FAST and COLOR
SLOW and COLOR
FAST and MONO
SLOW and MONO

FAST/SLOW is for very old CGA and MDA screens. Depending on the type of the display controller, white specks could be seen on the screen whenever the computer wrote data to the screen display. If you are experiencing this condition, you can change the display mode from FAST to SLOW.

This completes the hardware and software installation. If you are already familiar with the programmer, you may begin programming devices immediately. To get a better understanding of all of the commands and capabilities of the programmer you may want go through the tutorial in chapter 3.

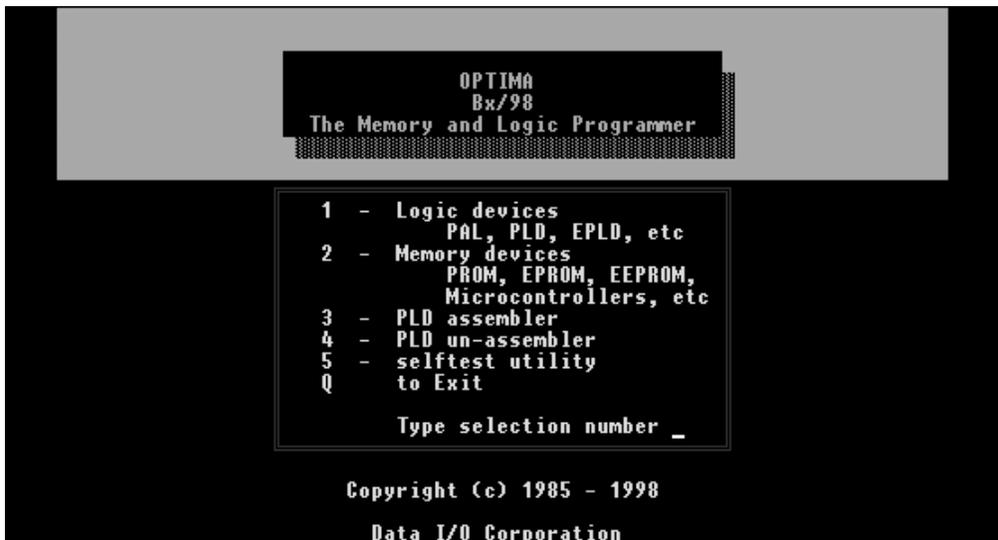


### 3. Optima Tutorial

This chapter presents a complete review of the programmer's features and capabilities. If you are already familiar with other programmer interfaces, you will find the user interface intuitive and very fast.

Once you are into the main menu, you will be able to execute commands by typing a single letter or number designator found in the menu. Single keystroke commands will be in brackets < >. If the instructions say enter <xx>, you merely need to press the key indicated within the brackets.

- 1) To start Optima DOS make sure you are in the right directory and type **SPRINT <ENTER>**
- 2) The operating menu should appear with the following menu options.



**Figure 3.1:** The Main Menu

**Note:** You may execute commands by simply typing the letter or number designator that appears in front of the menu item. If you type the designator, the command is automatically executed and you do not have to press <ENTER>. This makes getting around the menus very fast. For example, when the instructions say "type <1>" this is a single keystroke that will execute the desired command, in this case it will start the PAL driver.

**Note:** *PldAsm* and *UnAsm* are older utilities that are no longer supported.

Depending on the configuration you have ordered, the PLD assembler and UNASM products may not be included, if not, then the menu items 3 and 4 will be marked with (DEMO).

## Running the Programmer

The following sessions will take you through each of these functions and provide you a complete tour of the programmer's capabilities. We will be using the files loaded into a sub-directory called "Examples" that was installed in your Sprint directory during installation.

## Logic Devices

We are going to go through the commands for Logic Devices first. Many of the commands are the same for Logic and Memory. However, there are some differences, which will be covered, in separate sessions.

### Selecting a Device

Type <1> from the main menu and a list of all the PLD device manufactures will be displayed. You may choose a manufacturer by typing the designator for that manufacturer. If this is not the first time you use PAL, the manufacturer you selected last time will be indicated with an >.

For this session, we want you to select a PALCE22V10 from AMD. To do this type the letter designator you see in front of AMD. Because of the large number of devices from AMD, the PLDs are divided into sub-categories depending on the family of device. The screen will now show the list of AMD families of devices that are supported. (note, there is a separate list for MMI devices prior to their acquisition by AMD). The 22V10 is available in different technologies, but we will select the PALCE device today, so you will need to type the letter designator for PALCE and the screen will show a list of AMD PALCE devices. You may now select the 22V10 by typing the letter designator in front of the device. Note that in each step, Manufacturer, Family, Device, an arrow or Highlight indicated the device chosen in the previous session, if you were to have typed <SPACE> then the device selection would have been completed in one step.

Now that you have selected a specific device, the following command options appear on the menu. Continue the session and you will learn about the major capabilities enabled with these commands:

```

OPTIMA: Logic programmer Bx/98
▶ A = Assemble PLD to JEDEC file
  C = Change device type
  E = Edit pattern in memory
  G = Create and debug test vectors
  I = Read JEDEC file from disk to memory
  M = Multiple device programming
  N = Select multisyte unit
  P = Program device
  Q = Quit
  R = Read device
  S = Program security fuse
  T = Test device using JEDEC vectors
  U = Un-assemble memory to file
  V = Verify device to memory
  W = Write data to JEDEC file
  X = Cross programming utility
  D,H,0 : additional commands
          Dec 22 1998
Enter command _

Device code is 218535
AMD PALCE22V10H
Device type selected

0
Checksum

```

Figure 3.2: The Logic Menu

**Command <E> - Edit pattern in memory.**

Now we are going to look at the fusemap editor. Note the checksum displayed. Type **<E>** to enter the editor and the fusemap will be displayed. To see a list of commands available in the editor type **<F1>**. Unless you have loaded a file into memory or read a device into memory, the displayed fusemap will be for the blank device you selected. In this case it should be for the 22V10 device we selected at the beginning of this session.

You may go ahead and change bits in the fusemap in the display. When you are done, press the **<ESC>** key to exit back to the command menu. Notice that the checksum has changed based on the changes you made.

**Command <D> - Execute DOS command.**

This is a standard 'jump to DOS' routine found in many programs. Optima DOS will remain resident while you execute the DOS command. To get back to Optima DOS type **EXIT** from the DOS prompt. The device will have to be reselected when you return to Optima DOS, and any file loaded will have to be reloaded.

**Command <C> - Change device type.**

Entering **<C>** will allow you to select a different device. This command will take you back to the vendor selection menu. Go ahead and try it. You can now follow the same sequence we used at the opening of this session to select the device we are using in the exercises.

**Command <I> - Read JEDEC file from disk to memory**

With the **<I>** command we are going to load a file from the examples directory into memory and explore some of the features of the File Input Menu. Type **<I>** and the file-input menu will be displayed. Move the highlighted area with the cursor keys to select the **EXAMPLES** directory. Pressing the SCAN mode control (**<F2>** key) allows the files to be color coded according to their compatibility to the selected device. Notice that the DOS cursor is at the top of the screen, waiting for you to type in the name of the file we are going to read. You could type in the complete path and file name, then **<Enter>**. However, it is faster to use the arrow keys to find the file you want to input, with SCAN mode skipping devices that do not match.

All of the settings for drive, directory and file type will default to the last set-up. If this is the first time the file input has been used, the default will be the directory from which you powered up Optima DOS and the file type will be '\*.JED'. You can move the Highlight to the Drive or Directory or File type fields with the TAB key to change as you need.

The file we are going to input is located in the Examples sub-directory that was loaded when you installed Optima DOS onto your computer. Using the arrow keys, highlight the EXAMPLES directory and press **<Enter>**. Notice that the highlighter went directly to the 22V10.JED file. This file is compatible with the device we have selected and is the first compatible file found in this directory. If you have a color monitor, you will notice the color-coding used to help you quickly move to the right file. If there were more than one compatible file in the directory, they would all be yellow while the other files would be gray or green. If the first file the highlighter stopped on is not the right file, you can quickly move to the right file with the arrow keys. Since this is the file we want to input, press **<Enter>** and the file will be loaded into memory.

You should now be back at the main programming menu and you will notice on the screen that the Filename in Memory is the 22V10.JED file we just loaded.

You may now go back to the fusemap editor and see the contents of the file you just loaded. See the **<E>** command.

**Command <W> - Write data to JEDEC file.**

The <W> command will write the contents in memory back to a JEDEC file on the disk. Type <W> now and the screen will prompt you to enter the directory in which you wish to write the file. The default directory will be the last directory you set with the file-input command. In this case, let's write the T22V10.JED file back into the Examples directory. Press <Enter> to select this directory.

You will now be prompted to type in the name of your file. You can use any name up to 8 characters. You do not need to type an extension because Optima DOS will automatically append the extension .JED to your file. Go ahead and enter any name you like and press <Enter>.

Now go back to the File Input Menu (<I> command) and you will see the file you just created.

**Command <T> - Test device using JEDEC vectors.**

To view this command, you will need a programmed device and a file with test vectors. If you have a 22V10 (any manufacturer as long as you go back and select that manufacturer) you can download a file from the Examples directory called 22V10.JED. This file contains test vectors. Once you have the file in memory, enter <P> to program the device, and <Y> to confirm programming. Once you have a programmed device, you can proceed with this exercise.

Enter <T> to run the test vectors. If there is any error the display will show the vector number that failed, what it expected to see and then what it actually measured. By pressing any key, an error message will display a descriptor of the error.

You will be asked if you wish to debug these vectors with the G command. Entering <Y> will take you into the G command screen. This command is explained in detail elsewhere, so let's exit with the <Q> command.

**Command <R> - Read device.**

The read command will read the contents of a device into memory. Unless you are actually using a physical device in this exercise, you will not be able to execute this command; you will get a continuity error when the <R> (or <P>) command is executed. Remove any device from the socket, and enter <R> now, the programmer will attempt to read. Prior to reading, the programmer will test the pins of the device to make sure that contact is being made. If there is a continuity failure, the read function will fail and you will be able to see a diagram that indicates which pins failed. Go ahead and execute the read command and then enter <Y> to display the continuity errors.

Since this is an empty socket, all of the pins failed continuity. The pin numbers in parentheses (x) are the ground pins. The pins that failed will be displayed also. Since there was no device in the socket, all of the pins will be displayed as failed. If you were actually reading a device and got the failure, this feature can help identify which pin had the failure.

This error could indicate a misjustified or backwards device but, it also could indicate poor contact due to dirty pins on the device or dirty contactors in the socket. If you discover one of the problems, you can correct the condition and initiate the read command again by typing <Y> to retry.

Pressing <ESC> or <N> will abort the command and take you back in the main programming menu.

**Command <P> - Program device.**

The <P> command will initiate programming of the device. In order to prevent you from accidentally programming a device, after you press <P>, you will be prompted to confirm the operation by entering <Y>. Prior to actually applying the programming signals, the programmer will initiate a blank check if the device is not electrically erasable, otherwise the device is erased before programming. Since in our example the socket is empty; you will get the same continuity error we experienced in the read command <R> exercise.

In an actual programming operation, the programmer will proceed with programming after the blank check/erase cycle. It will then verify the programming by comparing the contents in memory with the actual data that is read back from the part, some device require verification at different voltages. The next step is to program the security bit if this function has been enabled and finally if test vectors are appended to the file it will then run the vectors.

**Command <M> - Multiple device programming**

You can program a set number of devices without typing <P Y> for every device. See the PROM section for an example of this command operation.

**Command <S> - Program security fuse**

Type <S> to enable the programming of the security fuse. You will have three options available with this command. If you enter <Y>, the security fuse of the device in the socket will be programmed now. When you use this command, keep in mind that the device should already be programmed. Bipolar and OTP devices will not be able to be reprogrammed once the security fuse has been programmed.

If you enter <A> the programming of the security fuse will be automatically enabled as part of the programming cycle. Go ahead and type <A> now and you will notice on the menu that next to the description for the <S> command the word (auto) will appear. This is to let you know that the automatic programming of the security fuse has been disabled.

Type the <S> command again to disable this function. Type <N> and the automatic programming of the security fuse will be disabled. Notice that the (auto) message is now gone.

**Command <V> - Verify device to memory.**

The <V> command will compare the contents in the device against the data in memory. If the contents do not match, you will be able to display an error map that will indicate the errors.

In the error map only the rows where failures occur will be displayed. For each location that failed, the display will indicate that location with either an 'X' or '-' or ' ', depending on what it expected to find. Note that the data displayed is what the programmer expected to find based on the contents in memory. For those locations where the device and memory were the same, neither X nor - is displayed.

## Memory Devices

In the next session, we will review features that are specific to Memory type devices. To begin this session you need to be in the start-up menu that comes up upon running Optima DOS. If you are still in the Command screen for PLDs, enter <Q> and you will be put back to the start-up menu.

### Selecting A Device

Type <2> from the main menu and a list of all the Memory and Micro device manufactures will be displayed. You may choose a manufacturer by typing the designator for that manufacturer.

For this session, you need to select an Intel 27C512. To do this, type the letter designator you see in front of Intel.

The Intel devices are divided into 4 categories, Flash, Micros, OTPs and EPROMs. For the 27C512 enter <D> and the screen will display a list of EPROMs. You may now select the 27C512 by typing the letter designator in front of the device.

Now that you have selected a manufactures specific part, the following command options appear on the menu. Continue the session and you will learn about the major capabilities enabled with these commands.

```

OPTIMA: Memory Programmer Bx/98
> A = Set address range
  B = Blank check device
  C = Change device type
  D = Execute DOS command
  E = Edit pattern in memory
  I = Input file from disk to memory
  L = List contents of memory
  M = Multiple device programming
  N = Select multisyte unit
  P = Program device
  Q = Quit
  R = Read device
  V = Verify device to memory
  W = Write HEX or BINARY disk file
  Z = Device serialization options
  F,H,O,S,X : additional commands
                Dec 22 1998
Enter command _

Device code is 628691
Intel 27C512
Device type selected

0000
Checksum

ALL BYTES
Data-Type

```

Figure 3.3: The PROM Menu

### Command <A> - Set address range

The <A> command will allow you to specify a range within the device that operations will be performed on. By entering <A>, you will first be prompted to indicate the starting address of the range. Type 0100 <Enter>. You will now be prompted to enter the stop address. Type 0FFF <Enter>. Notice on the screen, the range you selected is now indicated. Blank check, verify and program operations are performed only to the selected range of the device.

**Note:** The default range is equal to the size of the device selected.

**Command <B> - Blank check device.**

The <B> command will read the device to determine if it is blank. If you have an Intel 27C512, insert it into the socket and enter <B>. The screen will indicate pass or fail depending on the state of the device.

If you enter <B> with an empty socket, you will get a continuity error. You will be prompted to display the continuity errors or not and if you enter <Y> a diagram of the failed pins will appear on the screen. Only the pins that failed will be displayed. Since there was no device in the socket, all of the pins will be displayed as failed. If you were actually reading a device and got the failure, this feature can help identify where the failure occurred.

This error could indicate a misjustified or backwards device but, it also could indicate poor contact due to dirty pins on the device or dirty contactors in the socket. If you discover one of the problems, you can correct the condition and initiate the read command again by typing <Y> to retry.

Pressing <ESC> or <N> will abort the command and take you back in the main programming menu.

**Command <C> - Change device type.**

Entering <C> will allow you to select a different device. This command will take you back to the vendor selection menu. Go ahead and try it. You can now follow the same sequence we used at the opening of this session to select the device we are using in the exercises.

**Command <D> - Execute DOS command.**

This is a standard 'jump to DOS' routine found in many programs. Optima DOS will remain resident while you execute the DOS command. To get back to the user interface, type **EXIT** from the DOS prompt.

**Command <O> - Display programming values**

Enter <O> and a list of programming values for the device selected will be displayed. Most of these values are device specific and are determined by the device manufacturer.

Optima DOS allows some of these parameters to be temporarily changed by the user. To change a parameter move the cursor up or down the screen to the line that contains the parameter you want to change. Not all of the parameters shown may be modified. The cursor will stop only on the parameters that can be modified.

**Note:** *The variable parameter mode can be very useful for troubleshooting programming problems. However, it can also be dangerous to your devices if you set the parameters outside the device manufacturers specifications. Therefore, Optima DOS will not write new programming values to the disk. The values will be re-set to the factory default settings as specified by the device algorithm. You may set the parameters back to the default settings by re-selecting the device by using the <C> command or by exiting from the user interface and then restarting Optima DOS.*

**Command <E> - Edit pattern in memory.**

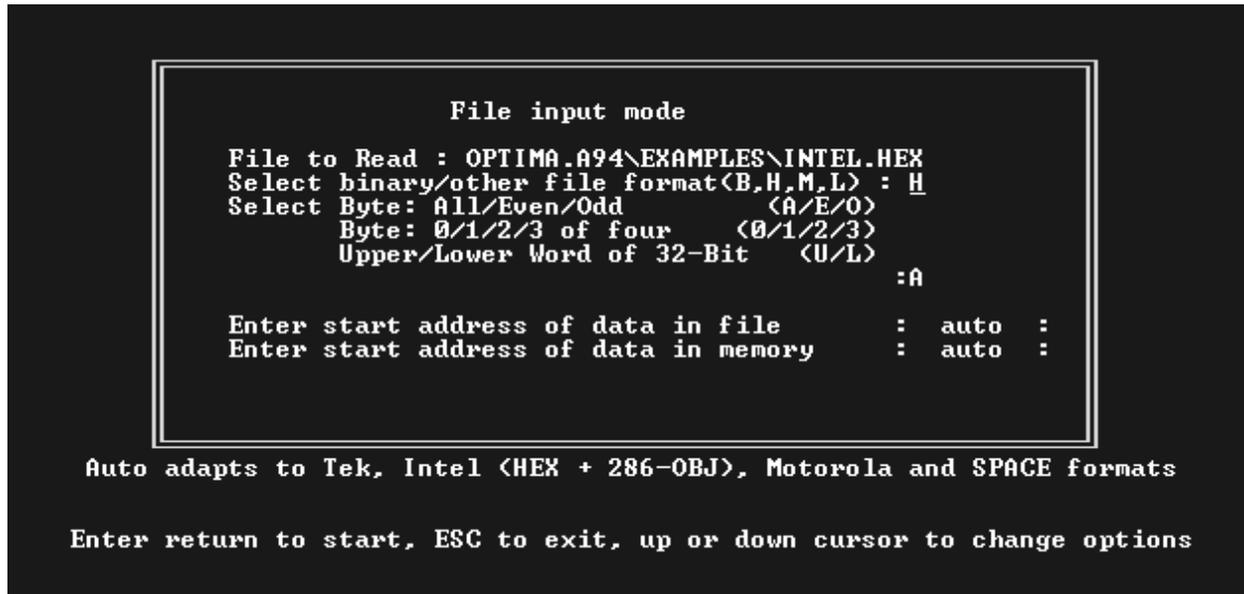
Enter <E> to view the edit mode. The contents of RAM will be displayed in HEX bytes and ASCII text. You may enter data in either format. To toggle between HEX and ASCII press the apostrophe <'> key. By pressing <F1> the help mode will give you a list of optional commands supported in the edit mode.

The contents of memory can be printed by using the <L> command.

**Command <I> - Input file from disk to memory.**

The <I> command allows you to input a file from disk into RAM. The data can be loaded as 8, 16 or 32 bit patterns; Odd/Even 16 bit patterns or U/L for 32 bit data patterns.

To view the Input command type <I>. Use the arrow keys to select the INTEL.HEX file. The File Input screen will appear and you will be prompted to identify the file type you have selected. The default format is Binary. For this exercise, your options are <B> for binary, <H> for Tek-Hex, Intel.Hex, Motorola.Hex or <M> for Mos Technology or ASCII Space. Type <H> for HEX format.



**Figure 3.4:** File Input Options

Using the "down" arrow key, go to the next parameter and you will be prompted to choose the bit pattern for the data you are inputting. Select one of the following options:

<u>command</u>	<u>function</u>	<u>File Data</u> <u>size</u>	<u>EPROM</u> <u>size</u>
A	All for 8 bit data	any	same
E	Even for even bytes of 16 bit data	16	8
O	Odd for odd bytes of 16 bit data	16	8
0	Byte 0 of 32 bit data	32	8
1	Byte 1 of 32 bit data	32	8
2	Byte 2 of 32 bit data	32	8
3	Byte 3 of 32 bit data	32	8
U	The last 2 bytes of 32 bit data	32	16
L	The first 2 bytes of 32 bit data	32	16

Select <A> for all 8-bit data. Press <Enter> and the file will be loaded.

**Command <L> - List contents of memory.**

Command <L> will allow you to list on the screen the contents RAM. Go ahead and enter <L> now. You can list any portion of the memory to the screen. The format you need to type is 'start address, stop address' <ENTER> and the contents will be displayed on the screen. You can also add an optional asterisk (\*) in the command line and the output will be directed to your printer port.

**Command <M> - Multiple device programming.**

The **<M>** command should be used if you want to program more than one device with the same data pattern. Go ahead and enter **<M>**. You will be prompted to enter the number of devices you want to program. Next you will be given the option of eliminating blank check. If you know that the devices you are using are blank, you may want to type **<N>** at this point to eliminate this test. For some of the very large Memory devices, this test takes some time, which is wasted if you already know that your parts are blank. A counter is then set and all you will need to do is press the **<space bar>** to initiate the next programming cycle. As soon as you reach the number of programmed devices you specified, the operation will stop. The counter will increment with good devices only. If a device fails to program, it will not be counted.

The **<ESC>** key will abort the **<M>** command.

**Command <P> - Program device.**

The **<P>** command will initiate programming of the device. In order to prevent you from accidentally programming a device, after you press **<P>**, you will be prompted to confirm the operation by entering **<Y>**. Prior to actually applying the programming signals, the programmer will initiate a blank check. If the device is not blank, the programmer will evaluate the contents to see if the data in the device can be successfully overprogrammed. If it can, the programmer will prompt you to initiate programming over the existing pattern. If it is not possible, you will get an error message and the operation will abort.

**Command <V> - Verify device to memory.**

The **<V>** command will compare the contents in the device against the data in memory. If the contents do not match, the failed addresses will be displayed along with the data in memory and the data that was read from the device.

**Command <W> - Write HEX or BINARY disk file.**

The **<W>** command will write the contents in memory back to an Intel-HEX, space-HEX or binary file on the disk. Type **<W>** now and the screen will prompt you to enter the format of the file you wish to write to the disk. Enter **<H>** for INTEL-HEX. Next you will be prompted to enter the directory path for the location where the file is to be stored. The default directory will be the last directory you set with the file-input command. In this case, let's write the INTEL.HEX file back into the Examples directory. Press **<Enter>** to select this directory.

You will now be prompted to type in the name of your file. You can use any name up to 8 characters. You do not need to type an extension because Optima DOS will automatically append the correct extension to your file. Go ahead and enter any name you like and press **<Enter>**.

Now go back to the File Input Menu (**<I>** command) and you will see the file you just created.

If you have made it this far, you should be familiar with all of the major features of the programmer and are ready to start programming.



## 4. Logic (PLD) Functions

This chapter will go through all of the functions and features enabled when you select Logic devices (item #1) from the opening menu, or start PAL from the DOS prompt with **PAL**.

### COMMAND LINE OPTIONS

It is possible to start PAL from the DOS prompt with a set of command line options. These are normally not required, but may be interesting for special applications. The options are:

- K     disable BEEP when programming is complete
- E     reload SPRINT main menu when exiting
- Fxx   use filename xx for remote mode, can include complete drive:\directory\file path
- Bxx   use filename xx for BATCH commands
- Q     select previously used device type and filename, go directly to command menu
- R     record keystrokes for making a BATCH file
- Dxx   select device code xx (6 digits) as initial device
- Sn    Force selection of base position #n, 1 is right-most position, then 2,3 etc  
(Only available for the Multisyte programmer.)

## DEVICE SELECTION

```

          PLD vendor selection with any TOP

# = my own list
A = Altera                Q = Ricoh
▶B = AMD                  R = Samsung
C = AMD/MMI              S = Seeq
D = Asahi Kasei          T = Philips/Signetics
E = Atmel                U = Philips
F = Cypress              V = Sprague
G = Exel                 W = SGS-Thomson
H = Gould/AMI            X = Texas Inst.
I = Hyundai              Y = Toshiba
J = ICT                  Z = Uantis
K = Intel                F1= VLSI Tech.
L = Lattice              F2= Wafer scale
M = National Semi       F3= Xilinx
N = PLX Tech.           F4= JAM
                        F5= Vector test only
      Plus Logic ->PROM
      Quick Logic ->PROM
Enter vendor code      SPACE: select last used, ESC: exit
                      OPTIMA Copyright (c) 1991 - 1998

          Data I/O Corporation
Press + to generate device list for PAL
Bx/98

```

Figure 4.1: PLD Vendor Selection Menu

After starting PAL you will enter the vendor selection menu and the screen will display a list of manufacturers supported by the programmer. Next to each manufacturer is a letter, number or function key that can be used to select a device from that manufacturer.

```

          PLD family table for vendor = AMD

A = Am -- std
▶B = PALCE --
C = PALLV --
D = MACH --
E = PALCE16V8 as --
F = PALCE20V8 as --

Enter family code _

```

Figure 4.2: A PLD Family Table

Enter the designator for the manufacturer of the device you are selecting and you will enter the next level in the device selection menu. Either a list of devices by the selected manufacturer will be displayed, or in many cases the list of devices is much too long for a single screen. In these cases, the list of devices will be subdivided into families. Select the required family if necessary.

```

          PLD device table for vendor = AMD

A = PALCE16V8H      M = PALCE20V8H      Y = PALCE22V10Z
B = PALCE16V8H/4   N = PALCE20V8H/4    Z = PALCE24V10H
C = PALCE16V8H/5   O = PALCE20V8H/5    F1 = PALCE26V12H
D = PALCE16V8Q     P = PALCE20V8Q      F2 = PALCE26V12H/4
E = PALCE16V8Q/4   Q = PALCE20V8Q/4    F3 = PALCE29M16H
F = PALCE16V8Q/5   R = PALCE20V8Q/5    F4 = PALCE29M16H/4
G = PALCE16V8Z     S = PALCE22V10H     F5 = PALCE29MA16H
H = PALCE16V8Z/4   T = PALCE22V10H/4   F6 = PALCE29MA16H/4
I = PALCE16V8HD    U = PALCE22V10H/5   F7 = PALCE610H
J = PALCE20RA10Z   V = PALCE22V10Q     F8 = PALCE630H
K = PALCE20RA10H   W = PALCE22V10Q/4
L = PALCE20RA10H/4 X = PALCE22V10Q/5

Enter Selection _

```

**Figure 4.3:** A PLD Device Table

Now select the device by entering the designated letter, number or function key associated with the specific device, or **<space>** if there are more than 36 devices in the family. After the device is selected, you will then be in the main programming menu.

The menu does not list device speeds or package types that do not affect the programming algorithm. All of the significant elements of the device name will be listed on the screen. For example, all Cypress PALC22V10B devices program the same way regardless of speed, so no speed is listed. However, the pinout of the LCC device is different than the PLCC device, so the LCC device is listed. AMD PAL16xx-7 devices program differently than the -10 devices, so here the speed is listed.

The size of the memory buffer is automatically set to the device size selected. Changing between compatible devices will not erase the data stored in memory. If a device type is changed to another device having an internal organization different from the initial device, the memory is reset to the blank condition in order to avoid operator errors.

PLDs have different internal sizes. Only valid fuse locations are enabled when a PLD type is selected. The internal size is compatible with the JEDEC data files generated by all PLD development software.

### BUILT-IN 'WALL CHART'

The user may optionally make a printout of all the devices supported. To do this, go to the vendor selection menu and type **<\*>** for a direct printout onto the printer, or **<+>** if you want to write the list to an ASCII file on a disk.

There are three options available when printing the device list. The first option is to choose the either 132 or 80 character/line mode. Optima DOS supports both the IBM standard (EPSON) command set for 132 character mode, and the HP PCL (printer control language).

The second option is to generate a long list with all detail (device size, number of pins, socket adapter required, GANG mode support etc). The list has about 2000 entries, and will take at least 20 pages of paper in 132 character/96 line mode. If this option is not chosen, the list will fit onto 3 pages, and will be specific to the type of socket installed, that is if a TOP40 is installed, only the devices supported on the TOP40 will be listed, likewise for TOP44, 68 and so on.

The third option is to print the unique ID code assigned to each device. This can be handy if you are creating your own device list or incorporating device selection in a batch command.

### MY OWN DEVICE LIST

You can create a device list that displays only the manufacturers and devices that you use. You can store up to 26 PROM devices and 26 PAL devices in this user defined screen. To create your user defined menu you will need to create a file called PRMDEV.CNF for your memory devices and PALDEV.CNF for your PAL devices. These files need to reside in the directory with PROM.EXE and PAL.EXE. To create the file simply create an ASCII string of device ID codes separated by a semicolon. The device ID codes can be found in the upper right hand box on the programming menu or you can print a list of devices with their associated ID codes by entering <+> or <\*> in the vendor selection menu.

e.g. The file PALDEV.CNF with the contents : 219579 ; 241379 ; 251423  
will create My Own Device List for the AMD MACH210, ICT PEEL18CV8 and Lattice pLS11032.

### MAIN PROGRAMMING MENU

Following the device selection, the main programming menu is displayed on the screen. This menu is composed of a list of commands. Command selection is performed with a single key stroke (the letter corresponding to the command in the menu). Commands may be selected with either upper case or lower case letters; both are accepted. The function of each command is described in the following paragraphs of this chapter. In the top right-hand corner of the screen the vendor, device and device ID code that have been chosen are displayed. If this does not correspond to the vendor or the device that is to be programmed, then the user may change the device type and the vendor by executing the command <C> (Change device type), see below.

### DISPLAY ERROR MAP

If an error occurs during Blank Check, Verify, or Test, the user will be prompted with the message **'ERROR - display detailed report? [y/N]'**. Entering any key except <Y> will cause the main menu to be displayed with the **'FAIL'** box blinking in inverse video. If the error display continues over 20 lines, the message **'MORE ? [y/N]'** will be displayed, enter <Y> to continue, any other key to exit.

For test errors, either during a programming cycle or from the main menu <T> command, the user will have the error displayed on the screen, and will be prompted to either display more errors, or exit. Type 'space bar' to have the failing test vector location graphically displayed on the screen. The expected data and actual data will both be displayed.

## PLD Programming Commands

```

OPTIMA: Logic programmer Bx/98
▶ A = Assemble PLD to JEDEC file
  C = Change device type
  E = Edit pattern in memory
  G = Create and debug test vectors
  I = Read JEDEC file from disk to memory
  M = Multiple device programming
  N = Select multisyte unit
  P = Program device
  Q = Quit
  R = Read device
  S = Program security fuse
  T = Test device using JEDEC vectors
  U = Un-assemble memory to file
  V = Verify device to memory
  W = Write data to JEDEC file
  X = Cross programming utility
  D,H,0 : additional commands
          Dec 22 1998
Enter command _
  
```

**Figure 4.4:** PLD PROGRAMMING COMMANDS MENU

**Note:** Do not select the <A> Assemble PLD to JEDEC file or <U> Un-assemble memory to file commands. They are now unsupported.

### <B> BLANK CHECK (non 'EE' type devices)

This command verifies that the device is blank. The device in the socket is read and the contents are compared to the expected 'erased' condition. The device type selected must match the physical device inserted in the socket. A blank check is not the same as reading a blank device, the programmer automatically uses the special sequences required.

If an error occurs, the user has the option to display an XPLOT of those fuses that are not blank. The plot on the display is an error map. Only the rows that are not blank will be displayed and the expected data will be displayed as an 'X' or '-', depending on the normal state of the fuses in a blank device.

Electrically erasable devices do not require a blank check operation. These devices are automatically erased as part of the programming cycle. Therefore, the Blank Check command does not appear in the menu. However, this command is still active if you want to execute a blank check on one of these device types.

### <C> CHANGE DEVICE TYPE

The active device type can be changed using this command. This command will take you back to the vendor selection menu described in the Device Selection section at the beginning of this chapter.

When the device type is changed, the active memory buffer area is automatically changed to accommodate larger or smaller devices as necessary. However, data previously stored in the memory buffer remain unchanged until either a device is read, data is read from the disk, or a device with a different internal architecture is selected. It is therefore possible to select one device, read it, change to a second source for that device, and program the new device without using disk I/O commands.

**Note:** Reading, Blank check and Programming cycles involve voltages above  $V_{cc}$  (3V or 5V depending on device), the device type selected must match the device in the socket. If an incorrect device is inserted in the socket, damage to that device will likely occur.

### CROSS PROGRAMMING

Cross programming is a feature provided to support many 16V8, 20V8 and related devices. The idea of cross programming is that the programmer accepts a first generation device type (for example 12H6) and programs a JEDEC file for that device directly into a replacement device (the 16V8). The JEDEC map of the 12H6 is not compatible to the 16V8, therefore the programmer must do the translation at the moment when the 16V8 is to be programmed. Optima DOS supports 42 different first generation devices, and presently 2 different replacement devices (16V8 and 20V8).

To use cross programming, select the replacement device listed in the Device Selection Menu with the 'AS' option. For example '16V8 as'. Then select the first generation device from the '16V8 as' menu (12H6). You can now read in the 12H6 JEDEC file and program the 16V8. All architecture bits and signature fields will be automatically configured. No translation is required.

### <D> DOS COMMAND

This allows MSDOS commands to be executed without leaving the user interface. Note that the MSDOS 'COMMAND.COM' file must be either in the SPRINT directory, or locatable via the DOS PATH command for this to work. If large PLDs are selected, there may not be enough room to load the MSDOS command file, the file stored in memory will be cleared before DOS is called, you will have to reselect the device type on completion. Once loaded, any normal DOS command can be entered, in the same manner as in DOS itself. For example:

**DIR \*.COM**

will list all the files with the extension or file type ".COM" (MSDOS binary files). Optima DOS will remain in the DOS command mode until 'EXIT' is typed on the keyboard. On completion of the DOS command, the user is returned to the main PAL utility menu.

### <E> EDIT PATTERN IN MEMORY

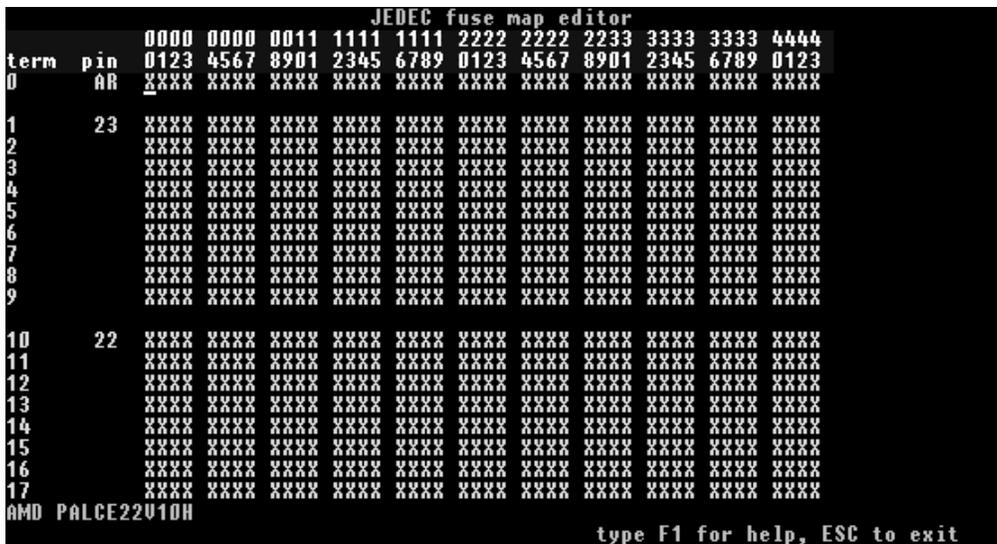


Figure 4.5: Editing Memory Contents

Optima DOS includes a very powerful editor to help you to understand the internal configuration of a PAL or EPLD. The display includes identification of the pin numbers associated with the various product terms, and displays the internal features with the node numbers corresponding to popular PLD compilers. The editor can be used to debug PAL designs, or to verify the output of assemblers or compilers. Any fuse in the device can be changed. We refer to the programmable elements of the device as 'fuses' even if they may be erasable CMOS cells.

After entering <E> for Edit, the user sees a display of the fuse map on the screen. By using the cursor keys, any fuse can be pointed to and changed if desired. For popular PAL and EPLD devices, the pin numbers corresponding to the product terms shown on the screen are displayed. Internal Set and Reset terms are displayed as SP and AR respectively. Device Architecture bits are displayed as 'A'. Other nodes internally are displayed as N1, N2 etc.

The Optima DOS then waits for user commands as follows:

home	goes to fuse 0
end	goes to first fuse, last product term or architecture row.
page up	24 lines up
page down	24 line down
arrows	moves cursor one location
^left	control left moves to left margin
^right	control right moves to right margin
<cr>	display next line
Backspace	moves cursor one location left
<esc>	exit to main menu
X	stores a 0, meaning 'connect' at the current fuse
-	stores a 1, meaning 'disconnect' at the current fuse
1	set entire product term to no connect (on)
0	set entire product term to all connected (off)
*	print the XPLOT on the printer
F1	display help screen
F2	do not display product terms that are off, (all '0')
F3	toggle screen mode - fast/slow
F4	display JEDEC fuse numbers/display product term numbers

After exiting, the new checksum of memory is calculated and displayed on the main menu screen. Any change in the checksum will cause the filename display to be reset.

### <G> CREATE AND DEBUG TEST VECTORS (OPTIONAL)

Please first read the Section on the 'T' function for more information about test vectors and related issues. This function is invoked in one of two ways. The 'G' command from the main menu will start the exercise directly. This is the method an engineer would use to develop and test a suite of test vectors. The 'G' command offers a graphic display of the device in the socket, and allows interactive development of PAL/EPLD vector test functions. Once a suite of vectors has been generated, they can be stored on the disk with the normal 'W' command - they will be attached to the JEDEC fuse description file. The optional UNASM software can also be used to create a transportable source file of the test vectors.

The second method is to optionally enter the exercise mode if the device in the socket fails the test vectors loaded into memory as part of the standard JEDEC file. If an error occurs, either as part of the normal programming cycle or from the main menu 'T' command, the user will be prompted with the option to 'enter test/edit mode'. In the production programming environment, the operator would enter 'N', and the menu will report 'FAIL.' In the laboratory, entering 'Y' to enter EXERCISE followed by 'SPACE' to run the vectors, will cause the failing location to be displayed on the screen, the failing bits are shown in red (or inverse video in monochrome systems).



This result can be stored as a vector for future use by entering either 'U' to update the current vector, or with 'M' to make a new vector. If a vector is made, it will be inserted into the vector sequence after the current vector location, which could be either the middle or the end of the vectors in memory. After updating or making a vector, the exerciser will run from the START vector (default is vector 1) up to the new vector. The reason for assigning a START vector, is that any registered PALs' present status is dependent on all previous states. An alternative to using vector 1 as START, is to set the start vector to one which preloads, resets, or otherwise assigns all registers inside the PAL to a defined state.

The execution of each test vector consists of 3 phases. First all inputs are set to 1 or 0 according to the vector. Then a clock pulse is applied to those pins with C or K levels. Finally, the actual device output pins are read and compared to the vector levels H, L, Z, . Any differences are displayed in red (inverse background on monochrome systems). If an error is detected during execution of the vectors, execution stops.

**Note:** The 'X' level should be don't-care. However many compilers expect 'X' to be logic 0, so Optima DOS applies a logic 0 to all pins that are not used as outputs by the vectors in memory. The possible output pins are allowed to float to a logic 1 level, allowing the output to pull the pin to 1 or 0 without causing a short circuit.

To save a set of test vectors on the disk, the user can quit the exerciser ('Q' or 'ESC') and store them on the disk with the main menu 'W' command. These vectors will be attached to the device fuse information and stored at the end of the standard JEDEC file.

## JEDEC Levels

Each pin can be set to any of the levels listed below. To set a level on one pin enter the level followed by the pin number. To set a range of two or more pin numbers, enter the level followed by the lower pin number, comma, then the higher pin number. With the range method, pins set as N (not used) or V (Vcc) will be skipped.

LEVEL	TYPE	FUNCTION
C	input	fast rise time clock from 0 to 1 to 0
F		float pin (same as '1')
K	input	fast rise time clock pin from 1 to 0 to 1
H	output	test pin for 1 output level
L	output	test pin for 0 output level
N		not used for I/O will float pin to 1 (unless used as power or ground)
P*		preload device outputs per device specs
B*		test buried registers
V		Set pin to 5 volt Vcc supply, see VCC section
X	don't care	will float to 1 or 0 depending on intelligent check of I/O usage
Z	output	test pin for floating condition
?	output	read any output level
1	input	set pin to logic 1 via 4.7K ohms (slow rise time)
0	input	set pin to logic 0, minimum 16 ma sink.

## Manually setting levels

After entering a LEVEL,PIN combination, the levels are first set in memory and displayed on the screen. They will be applied to the device only after the next single cycle command (simple **<Enter>**). In this manner, multiple new levels can be set as inputs before any clocks are applied to generate the next state in registered devices. As a reminder that the inputs on the screen are not yet applied to the device, the color of the vectors change to blue from green, then back to green (or red) after the single cycle. As long as levels are being changed, single step does not advance the vector pointer. If no levels are changed, single step executes the next vector unless none are loaded or the pointer is at the end.

**Note:** *The use of preload (**B** or **P**) implies a detailed knowledge of the device. This function is not supported by all devices, and it uses voltages above +5 volts according to each manufacturers specifications. The device in the socket must match the device chosen or damage may result. Optima DOS knows the preload procedure for all supported PALs. If preload is used, then the outputs become inputs (set as 1 or 0) and the data on the output pins is loaded into the /Q (Q bar) of the flip flop on that pin. **P** and **B** levels must be on pin one. Be sure that asynchronous resets are not activated by the transition in and out of preload, or the preload cycle will appear to fail.*

The sequence of the application of the clock is not defined by JEDEC. To generate portable test vectors, if a device has more than one clock input, then separate vectors must be used to assure the correct clock sequence (if critical). If the C or K level is set on two device pins, either one may be toggled first depending on programming system manufacturer. Optima DOS toggles pin 1 last.

**Note:** *High speed PALs will create noise in many systems. Thus the C and K levels employ active pull-ups to 5 volts, while the 1 level is a slower resistor pull-up. For this reason you may witness miscounting if 0 to 1 levels are used as a clock instead of C or K. Also, the K is preferred over C since the noise margin between cycles on K is better than C. (3 volts vs. .6 volts in TTL compatible devices).*

## VCC Drivers

Not all pins have VCC drivers available. You should refer to the section of Chapter 3 for the socket type you are currently using. While jumpers could be used to provide power to pins that do not have a built-in VCC driver, caution is required to avoid a short between a pin at '0' and the VCC level.

The 'V' level is not a JEDEC level, and is used only to indicate which pin is VCC. The JEDEC standard indicates this pin as 'N' - not used. Based on the device selected, Optima DOS will automatically provide VCC power and ground to all necessary pins of the device.

## FUNCTIONS

These are debug like commands for generating, single stepping, and editing test vectors. Initially the starting address (START) for vector execution is set to vector 1. This START address is always displayed in the screen. The functions are listed here. If a function has optional parameters, the user will be prompted to enter these options after the function code. If no option is entered, then the default is the value '1'. The escape key can be used at any time to abort the command. These function are listed on the following page.

<u>Key</u>	<u>Option</u>	<u>EFFECT</u>
	(number of option parameters)	

A	1-16	Assign state pin sequence. It is often useful to display the state of the device in a HEX number. With this command the pins of the device can be entered to be displayed in HEX on the screen after each step. Enter the pin corresponding to the LSB first
---	------	---

<u>Key</u>	<u>Option</u>	<u>EFFECT</u>
E	1	Execute. The option parameter becomes the START vector number, execution continues from start to end or until an error is detected. This new START vector will be used for future operations.  <b>Note:</b> <i>If the new starting vector does not set or reset (or preload) internal registers to a known state, errors may result.</i>
G	1	Execute. Runs from START, stops at vector number entered as the option.
I	3	Edit. Points to currently active vector on the right side of the screen. With the cursor keys, the contents of any vector can be changed. Press <b>&lt;Enter&gt;</b> when finished. The vectors in memory will be executed starting from the 'start' vector.
M		Make a new vector and Execute. The levels and actual device outputs will be written to a new vector inserted after the current location. Execute from START.
Q		Quit
R	1	Run. Execute the number of steps in option.
S	1	Mark end Sets the vector number chosen as the last vector.
U		Update and Execute Replace the present vector with the actual device levels. Pins with levels H, L and ? will have the actual device outputs set into the vector. Execution then runs from START until the end or until an error occurs.
<Enter>		Single step cycle. If no levels have changed, then step one cycle from the vectors. If any levels have changed, display the results of these manual level settings.
<space>		Run Execute from the current vector until the end, or until an error is detected.

## Execution Speed Considerations

Execution speed of the vectors is primarily dependent on the writing speed to the screen. For this reason, the user can specify to the Optima DOS PAL software, the type of screen attached. IBM MDA and Hercules cards will automatically operate at full speed. CGA and EGA cards may run at a slower speed. The CGA and EGA screens can be switched into fast display mode by setting the snow control flag OFF. The snow control flag may be toggled by the F3 key in EDIT. Once this snow control is set, it is stored in the JED\$\$\$\$.CNF file. With some CGA boards, the fast mode option will cause bits of 'snow' to be seen on the screen. If this is not desired, the slow mode can be selected, also with the F3 key in EDIT. The fast mode also improves all screen menu displays.

### <I> READ JEDEC FILE FROM DISK INTO MEMORY

```

Enter filename to read:_
Drive C:  Directory \EXPERT.A94\EXAMPLES      File type *.*
..
24U10.PLD      18U10.PLD      20RA10.PLD      22U10.PLD
7C331.PLD      26CU12.PLD     26U12.PLD       3T08DMUX.PLD
E16P4.PLD      7C332.PLD     DEMO.COM        DEMO.HEX
MAX20.POF      GAL6001.JED    GAL6001.PLD     INTEL.HEX
MAX7032.POF    MAX28.POF     MAX44.POF       MAX68.POF
PLX448.PLD     MAX84.POF     MOTOROLA.HEX    PLS153.PLD
TEP900.PLD     T16R4.PLD     T18N8.PLD       TEP600U.PLD
                TS26CU12.PLD  TST16R4.JED    VECTOR.PLD

F1: HELP      F2: SCAN
or use cursor keys to highlight item, then press Enter to select

```

**Figure 4.7:** Loading an input file from Disk

The <I> command loads the data from the file selected into the memory buffer. The system will automatically default to the last drive, directory and file type (extension) that was used. All files with the designated extension will be displayed. You may type in the file you want to input or use the arrow keys to select a file or change any of the defaults. For more details refer to the File Input Mode in the introduction section of this manual.

Optima DOS expects standard JEDEC format or POF format information. The JEDEC file is generated by any PLD development system, for example PALASM, ABEL, CUPL, LOG/iC, PLDASM, AMAZE, PLAN, PLDesigner, etc. The file contains all the data necessary to program the device for the desired function. If the data file contains test vectors, then these will be stored in a separate buffer for use by the, <T>, <G> and <P> commands. Typically 2300 vectors (28 pin devices) can be stored in memory, depending on system memory capacity.

Optima DOS checks and displays errors in the JEDEC file, device errors are

- a) the number of fuses,
- b) pin count and
- c) data checksum
- d) file checksum.

If a device has a different number of pins between the DIP and PLCC package types, Optima DOS will automatically adapter the test vectors to the number of pins in the physical package.

The checksum and the filename will be displayed on the main menu screen. A zero checksum indicates that no file is loaded.

If the JEDEC file contains the command G1, then the security fuse will automatically be set after programming, the word (auto) in the main menu indicates that security is enabled.

POF files are Altera specific, and are read in according to the Altera specifications.

### **<N> SELECT MULTISYTE UNIT UNIT (appears only when a Multisyte programmer is installed! )**

If a Multisyte system is installed, then the 'N' command can be used to :

- a) Select a different TOP, numbered 1 to 8, left to right. The device menu will appear with the device list that matches the installed TOP. The **<SPACE>** key can be used to return to the main menu without changing the device selection. Note that the device lists vary according to the installed TOP.
- b) Select all TOPs in parallel that have the same type as the previous selected. if the response to the 'N' command is a '0', then all units will be selected. This option allows from 2 to 8 devices to be programmed at the same time - with no difference in the programming cycle time. Each device has its own complete set of pin drivers.

Multisyte GANG mode increments the COPY counter by the number of units programmed. Importantly, GANG mode handles all error conditions in case a device fails in any operation. Any sockets that are empty during the programming will be ignored. Depending on device type and the operation executed, sockets containing devices with errors will be disabled for the rest of the operation, and indicated with a red LED. The verify and blank check operations will execute on all devices at once, if the devices do not have the same contents, then the cycle will repeat automatically to identify the defective devices, marking the devices with a RED or GREEN LED.

Read Device uses unit #1, otherwise all operations occur on all devices in parallel, empty sockets are skipped.

### **<M> MULTIPLE DEVICE PROGRAMMING**

This command sets up a counter for programming a fixed number of devices. After entering the number of devices to program press **<Enter>**. From this point on the operator need only insert the device, and type any key to program. Programming stops after the unit count is reached. Only successful programming operations will increment the counter.

**<O> DISPLAY SYSTEM CONFIGURATION**

```

Installed POD type is      0
Hardware is model        OPTIMA
Selected device          AMD PALCE22V10H
Device size              5828 fuses
Package size             28 pins

Screen display mode      ▶ Color / fast
Beep on programming done  0n
Command 'M' is configured with
                          Blankcheck
                          and Program
                          and Verify

Continuity check         0n
Vector test              0n

```

**Figure 4.8:** Display System Configuration

This command is used to display additional information about the system configuration. The type of socket installed (POD type) is shown on the first line. The type of hardware (Optima or PLUS48) is displayed, along with the device size, the pins in the standard package (DIP equivalent pin count is shown if the device is PLCC, and has a DIP version). The standard Vpp level (only for general reference, the exact level depends on the device ID). The display mode can be edited, as well as the options for BEEP, (one beep GOOD, two beeps BAD).

The continuity check can be disabled (NOT RECOMMENDED!) and the vector test can be disabled. Pressing the **<SPACE>** key will change the state of each of these options.

**<P> PROGRAM DEVICE**

When **<P>** for Program is entered, the device in the socket will be programmed using the data stored in the memory buffer. The device type selected by the user must match the physical device in the socket. Before programming, the user is first prompted to confirm the programming of the device by typing a **<Y>**, any other key will abort the programming command.

The device will first be checked for erasure (Blank check), unless the device is Electrically Erasable, then an erase cycle will be automatically executed.

If the device is not electrically erasable, Optima DOS will test if the pattern in memory can be over-programmed on top of the existing pattern in the device. If this is possible, the user will be prompted to **skip blank check**. Entering **<Y>** will over-program the device.

The device will be programmed using the algorithm defined by the manufacturer. After programming, the device will be verified to confirm proper programming. If there are any errors in verification, the user can have a detailed report on the screen. If the chip vendor specifies it, verify will be done at the Vcc high and Vcc low levels according to the data sheet. If the security fuse has been enabled, it will be programmed after verification. Any test vectors appended to the file will automatically be run as the last operation of the Program command.

The user is informed of the sequence of steps in the programming procedure by messages on the screen. A cycle counter in the lower right-hand corner of the screen will display the number of successful programming cycles completed with the data contents in memory. This counter is re-set by a read, input or change device command (unless the new device is compatible to the old one).

#### **<Q> QUIT**

This command returns the user to the Optima DOS opening menu.

#### **<R> READ DEVICE**

The **<R>** command reads the contents of the device in the socket into the memory buffer. Memory is first cleared, then the data is read-in. After data has been read into memory, the device type may be changed to a compatible device without altering any data stored. The device in the socket must match the device type selected or damage may result to the device due to the high voltages used to access the internal fuse structure.

If the device appears to be empty, Optima DOS will report a warning on the screen. A device may appear blank if its security fuse has been set.

#### **<S> PROGRAM SECURITY FUSE FUSE**

Many PLDs have an option to program some final fuses after all other fuses have been programmed. These final fuses prohibit reading (and verification) of the device. This means that the pattern set into the fuses of the device are no longer readable by those who should not have access to the pattern. These final fuses are called security fuses.

In manual mode, the user would enter **<S>** followed by **<Y>** to confirm the action. This must be the last step in any programming sequence since the **<P>** command expects to be able to verify (read) a device after programming.

In automatic mode the security fuse will be set during each programming cycle, after verifying and before vector testing. If the JEDEC file contains the security fuse command (G1\*), then automatic mode is enabled and main menu will display **Security AUTO**.

The automatic mode can be turned on from the keyboard by entering **<S>** followed by the **<A>** response, or turned off with the **<N>** response.

Optima DOS first programs the security fuse, then reads the array to verify that the data no longer matches the array. The PASS message is displayed only if the array data is no longer readable.

Some devices disable preload if the security fuse is set. For these devices Optima DOS will run the test vector program before setting the security fuse in the **<P>** programming cycle.

Some devices have a mode to read-out the security fuse bit. Optima DOS can read this bit and will indicate that the security fuse bit is set, however it will then perform a double check of the array contents to assure protection.

### **<T> TEST DEVICE USING JEDEC VECTORS**

If a JEDEC file with test vectors has been read into memory, then the device in the socket can be tested for conforming to these test sequences with this command. Entering the **<T>** for Test command will begin this test. This test will also be invoked after the verify operation in a programming cycle, if test vectors are in memory.

The Test operation has 3 passes for each vector. Pass 1 sets up all input pins. Any 0, 1, or X levels are presented to the device. All device outputs are set to read mode. Optima DOS has a special method for handling the 'X' (don't care) level, which is incompletely defined in the JEDEC specification and often is a cause of vector test failures in other systems.

Pass 2 applies any clocks to the device. The vector is scanned from highest pin to lowest pin, and the clocks are output in that order. If a multiple clock device requires a controlled sequence of clocks, then there should be a separate vector for each clock. It is impossible for a universal programmer like the Optima to generate simultaneous clocks on 2 pins. There will always be a time spacing between multiple clocks on in a test vector. Note that the 'C' clock is normally low, and the 'K' clock is normally high.

Pass 2 also executes the preload function if a P or B vector is found. A P or B in pin1 of the vector indicates preload. Note that entering and leaving preload mode can generate transient logic levels that can have other effects in a PAL design.

Pass 3 verifies the outputs of the device against the test vector. If there are no errors, the next vector is processed. If there are any errors, the good and bad data will be displayed on the screen, and testing will stop unless continued by the operator.

Pins defined as 'X' in the vectors are set as inputs to '0' unless the pin is determined to be an unused output, in which case it is floated. High impedance outputs are tested for incorrect '0' conditions, but not for low impedance '1' conditions. Test simulation can also be used to verify a device after the security fuse has been programmed,

**Note:** *If you experience problems with test vectors, please refer to the Trouble Shooting section of this chapter for more details.*

### **<V> VERIFY DEVICE TO MEMORY**

Compares the contents of the device in the socket to the data in memory buffer. Any differences are displayed in clear text, with memory data, device data and address. All features of the device are verified, including extra reset and enable bits and architecture bits.

**Note:** *The verify command will not function correctly if the security fuse of the device has been programmed.*

### **<W> WRITE DATA TO JEDEC FILE**

Writes the pattern and test vectors in memory onto the disk using the filename supplied. After entering **<W>** for Write, the user will be prompted to enter a DOS style path. A drive label, or path may optionally be entered. The default path is accepted if **<Enter>** is pressed without any other character. Then the filename may be entered. If another file with the same name already exists, the user will be prompted to confirm erasure of the other file.

**<X> CROSS PROGRAMMING UTILITY - PAL EMULATION (Optional)**

To use the cross programming utility, a device is selected from the device list, then is loaded into memory - either from a JEDEC file or from a device in the socket. This device may now be converted to another pin-compatible device with the X command. After typing X, PAL will store the file in a temporary source file. At this point, the user will be asked to select the new device. Finally the PLDASM will convert this source file into a new JEDEC file and load it into memory. Any test vectors in the original file will also be converted. It is suggested that the user store the converted file into the filename of his choice with the W command.

Any errors will be reported on the screen. A common error is that the destination device is unable to emulate the source device, such as if a non-inverted output is used and the destination device only has inverted outputs. The destination device must also have the same number of pins as the source device. All errors will be clearly shown on the screen.

**PLD Error Messages**

This is a summary of common error messages and how they should be managed.

**1) Error: display detail report? <Y/N>**

This message is output by blank check or verify. It indicates that the data read from the part did not match the expected data. This means that the device was not empty (blank check) or is not programmed correctly (verify).

**2) Skip blank check? <N>**

If a device was not blank during a programming cycle, it is possible to continue programming by entering <Y>. If you get this message during blank check, it means that Optima DOS has compared the device to the contents and RAM and determined that it is possible to program the new pattern into the device.

**3) Programming device - operation failed**

The device could not be programmed. Present silicon technology allows a 99% or better average programming yield. Depending on the manufacturing lot, the yield you experience may vary.

A common cause of programming failure is with erasable devices, which have been damaged in the application. If a device is removed from a PC board, erased, and then programmed, there should be no problem. However, if the device was damaged in the PCB, programming will fail. If the device is UV erasable, be sure that the device was exposed long enough to assure erasure.

**4) Manufacturers Electronic Signature incorrect, Silicon ID error**

Silicon Signature allows Optima DOS to adapt the programming voltages and pulse-widths automatically to a code provided by the vendor. The vendor provides specifications to Optima DOS before the silicon is sold to customers and we include these codes into the software. There are two possible causes for this message:

- a) The device has been damaged and is no longer programmable. Some CMOS devices are sensitive to latch-up and static discharge. This is a major cause of failures in CMOS products.
- b) The device is a new revision not supported by your version of Optima DOS. Contact your Optima distributor for an update.

**5) Preload not supported**

The device type selected does not support preload of test vectors.

**6) Test Vectors not supported**

This message occurs when an adapter is used in the 48-pin DIP socket and the pin count of the device is greater than 48 pins.

## Troubleshooting Test Vector Errors

Below is a list of common sources for test vector errors and workarounds.

**1) PRELOAD**

Preload is silicon device dependent. The same device type from different vendors use different preload sequences. Some of these sequences can interfere with normal device operation.

For example, if a Cypress 22V10 application uses pin 8 as an asynchronous reset input, then each time preload is done, the registers will be reset and the preload will fail. Other similar conditions exist will almost all devices. Use of the Optima DOS vector editor will allow these problems to be quickly identified.

If preload of the output registers is attempted in a device that does not support preload, no high voltages will be applied, and an error message will be reported.

**Note:** *As preload involves voltages above 5V, the device type selected must match the device in the socket. If an incorrect device is inserted in the socket, damage to that device may occur.*

**2) Power-on Reset**

If the first line of a sequence of test vectors fails, check that a power-on condition is defined in the equations. It may be required to add an initialization test vector to reset/preset the device registers as the first vector.

**3) Synchronous Clocks**

Under certain conditions, it is possible that a device could double clock. Double clocking means that a counter or state machine advances 2 steps instead of one step. This can occur if the device Tco is faster than the clock rise time, and multiple outputs switch at the same time, causing internal ground bounce. First, be sure that the fast clock drivers are being used (with the JEDEC 'C' or 'K' commands - not 010 sequences). PLUS48 clock drivers are designed to handle devices as fast as 5 ns without double clocking. The solution to this problem is to disable the outputs in the vector with the clock, then re-enable the outputs and complete the test in the next vector.

**4) Asynchronous or Multiple Clocks**

Clocking of devices with internal asynchronous clocks enabled (EP600, ATV750, ATV2500, 20RA10 etc) from pins with slow rise times may result in double clocking. This is application-dependant, and exists only in extreme cases. The easiest solution is to disable the outputs before clocking the device.

If a device has multiple clocks, it is not possible to drive these at the same time, there will always be some hundreds of nanoseconds of delay between clocks. Use 1 vector for each clock edge in multiple clock devices.

## 5) Signal Sequence

Pass 1 applies levels from pin 1 to the last pin of the device. Pass 2 applies clocks from the last pin backward to pin 1. Various equations are possible that may be affected by this sequence, for example a cross-coupled SET/RESET flip-flop is possible in a 16L8 device, this could malfunction under certain vector sequences. Make sure that all test vectors are sequence independent.

## 6) ATVG (Automatic Test Vector Generation software)

This software works from the JEDEC file, and appends to the JEDEC file test vectors. After each use of the ATVG software, the test vectors must be verified on physical devices, to eliminate the 5 effects listed above. In particular, some ATVG packages do not correctly handle multiple clocks. No ATVG package will manage the Preload interference problem.

## JEDEC File Standards

Optima DOS will accept JEDEC files from any development tool as long as the tool follows the JEDEC defined standards.

Fn\* sets the reset state of the fuses to n (either 1 or 0)

QFn\* defines the total number of fuses to be n

Gn\* enables the security fuse to be programmed if n = 1

Ln\* defines the fuses starting at fuse number n to the condition y (1 or 0). Sequential fuse locations are defined by a string of 'Y's until the terminating '\*' is found.

Cn\* the checksum of the fuse data is n

Vny\* defines the vector test number n to be the sequence y. A test vector contains the following letters (the pin sequence starts at device pin 1)

C = positive edge clock

K = negative edge clock

1 = set pin to > 2.4 v

0 = set pin to < 0.8 v

Z = test pin for high impedance

H = test pin for > 2.4 v

L = test pin for < 0.8 v

N = pin not used

P = test vector is preload condition

B = preload/test buried registers

An example of a program for a device with test vectors and preload is provided on the distribution disk in the file "TST16R4.JED"

## Device Specific Information

1) AMD PALCE devices disable preload when security is set., thus test vectors are executed prior to setting the security bit in the programming cycle.

## 2) MAX AND POF FILE FORMAT:

The MAX devices from Altera and Cypress use a new high-density format for the storage of the data previously stored in JEDEC format files. This new format, called POF, has two main concepts:

- a) High density - files are 6-8 times smaller than JEDEC files (for example an EPM5128 takes only 25600 bytes, not 200k bytes)
- b) Protection of the EPLD function. The data in a JEDEC file can be easily converted to and from Assembler source. The POF file cannot. It is used by Altera to isolate the internal operations of the device from the user.

Because of the complexity of the MAX devices, especially the 64 and larger macrocell versions, if you attempt to read or blank-check a device, you must have a POF file for that size of device (any POF file is OK) in memory. The device internal block structure is defined in the POF file. You will have an error message if no POF-file matching the device-size has been loaded.

## 3) ERROR DISPLAY FOR MAX DEVICES

To display errors from MAX-parts resulting from blank check or verify operations, special symbols are used to represent the 8 bits in each position. These graphic symbols indicate:

symbol number of bits at '1' in byte

X	0
i	1
l	2
-	3
-	4
"	5
—	6
¬	7
.	8

## 5. Memory Device Functions

This chapter will go through all of the functions and features enabled when you select Memory devices (item #2) from the opening menu, or start PROM from the DOS prompt with **PROM**.

### COMMAND LINE OPTIONS

It is possible to start PROM from the DOS prompt with a set of command line options. These are normally not required, but may be interesting for special applications. The options are:

- K      disable BEEP when programming is complete
- E      reload Optima DOS main menu when exiting
- Fxx    use filename xx for remote mode, can include complete drive:\directory\file path
- Bxx    use filename xx for BATCH commands
- Q      select previously used device type and filename, go directly to command menu
- R      record keystrokes for making a BATCH file
- Dxx    select device code xx (6 digits) as initial device
- Sn     Force selection of base position #n, 1 is right-most position, then 2,3 etc  
(Only available when using a Multisyte programmer)

## DEVICE SELECTION

```

PROM vendor selection: Page 1 with any TOP
# = my own list
A = Auto EPROM ID      P = LG Semicon      F5 = TEMIC
B = Actel              Q = Macronix        F6 = Texas Inst
C = AMD                R = Microchip      F7 = Toshiba
D = Atmel              S = Mitsubishi     F8 = WaferScale
E = AT&T              T = Motorola       F9 = Winbond
F = Catalyst          U = National       F10 = Xicor
G = Cypress           V = NEC            0 = Xilinx
H = Dallas            W = OKI            1 = Zilog
I = Exel              X = Philips/Signet
J = FAIRCHILD         Y = Samsung        PgUp: other vendors
K = Fujitsu           Z = Seeg
L = Hitachi           F1 = Seiko Epson
M = ICT               F2 = Siemens
N = Intel             F3 = SGS-Thomson
O = ISSI              F4 = SST

Enter vendor code _   SPACE: select last used, ESC: exit

OPTIMA Copyright (c) 1991 - 1998

Data I/O Corporation
Press + to generate device list for Memories + Micros Bx/98

```

**Figure 5.1:** The PROM Vendor Selection Menu

After starting PROM, you will enter the vendor selection menu and the screen will display a list of manufacturers supported by the programmer. Next to each manufacturer is a letter, number or function key that can be used to select a device from that manufacturer.

Enter the designator for the manufacturer of the device you are selecting and you will enter the next level in the device selection menu. Either a list of devices by the selected manufacturer will be displayed, or in many cases the list of devices is much too long for a single screen. In these cases, the list of devices will be subdivided into families. Select the required family if necessary.

```

PROM family table for vendor = AMD

A = Am27S--
B = Am--
C = Am27--
D = Am29SL-- B
E = Am29-- B
F = Am29LV--
G = Am29LV-- B
H = Am87--

Enter family code _

```

**Figure 5.2:** The AMD PROM Family Table

In this example, (AMD) you see the beginning of the device name, if the device name begins with 27S (a bipolar PROM) enter A, if it begins with 27 but not followed by S (it is an EPROM) enter C, otherwise enter B (Flash EPROMS, Microcontrollers, etc).

```

PROM device table for vendor = AMD

A = Am28F256          N = Am29F002NT
B = Am28F256A        O = Am29F002NB
C = Am28F512         P = Am29F010
D = Am28F512A        Q = Am29F100TE
E = Am28F010         R = Am29F100TS
F = Am28F010A        S = Am29F100BE
G = Am28F020         T = Am29F100BS
H = Am28F020A        U = Am29F200TE
I = Am2864AE         V = Am29F200TS
J = Am2864BE         W = Am29F200BE
K = Am2864B          X = Am29F200BS
L = Am29F002T        Y = Am29F200ATE
M = Am29F002B        Z = Am29F200ATS

type 'space' for more devices

Enter Selection _

```

**Figure 5.3:** A Vendor Device List Table

After the vendor and family menus, a list of devices by the selected manufacturer will be displayed. In some cases the list of devices is much too long for a single screen. In these cases, enter a **<space>** key to display the next screen of devices.

Select the device by entering the designated letter, number or function key associated with the specific device. After the device is selected, you will then be in the main programming menu.

The working size of the memory buffer is automatically set to the device size selected. The first time any device is selected, the memory buffer is initialized to the value set by the 'default buffer contents flag' (see Options). The default buffer size is 128 Kbytes; larger memory buffers are allocated to devices that require more than 128 Kbytes to store the entire device contents. If your PC does not have enough memory installed, an error message will be displayed. A checksum of the active area in the memory buffer is calculated and displayed on the main menu screen. For large devices, like the 27020, checksum calculation may take a few seconds. If the device size is larger than 2 megabits, Optima DOS will use the disk as storage. Optima DOS does not require any EMS or EXPANDED memory.

Some special memory devices have, in addition to the data array, additional control (reset or enable) or security functions. Optima DOS automatically adjusts the total memory size to include these extra functions. Those memory or microcontrollers that support a security fuse, an extra byte is assigned to control this function. These extra features, such as security fuses, initialize bytes, enable options etc, are always set to the default value when a device is selected.

## AUTOMATIC DEVICE SELECTION

In order to auto-configure the PROM programming utility to match the EPROM currently inserted in the socket, the user may optionally type an **<A>**. This will cause the PROM utility to attempt to read the device in the socket using 'silicon signature', determine its type and to configure itself for that device. If any error occurs in attempting to read the device in the socket due to no device in the socket or a device that does not support silicon signature, no auto-configuring will take place.

**Note:** *silicon signature must only be used with EPROMs, and: not all EPROM ID codes are unique, some ID codes have been used by two different vendors. Always confirm the Auto ID selection.*

## BUILT-IN 'WALL CHART'

The user may optionally make a printout of all the devices supported. To do this, go to the vendor selection menu and type `<*>` for a direct printout or `<+>` if you want to write the list to an ASCII file.

There are three options available when printing the device list. The first option is to choose the either 132 or 80 character/line mode. Optima DOS supports both the IBM standard (EPSON) command set for 132 character mode, and the HP PCL (printer control language).

The second option is to generate a long list with all detail (device size, number of pins, socket adapter required, GANG mode support etc). The list has about 2000 entries, and will take at least 20 pages of paper in 132 character/96 line mode. If this option is not chosen, the list will fit onto 3 pages, and will be specific to the type of socket installed, that is if a TOP40 is installed, only the devices supported on the TOP40 will be listed, likewise for TOP44, TOP1 and so on.

The third option is to print the unique ID code assigned to each device. This can be handy if you are creating your own device list or incorporating device selection in a batch command.

## MY OWN DEVICE LIST

You can create a device list that displays only the manufacturers and devices that you use. You can store up to 26 PROM devices and 26 PAL devices in this user defined screen. To create your user defined menu you will need to create a file called PRMDEV.CNF for your memory devices and PALDEV.CNF for your PAL devices. These files need to reside in the directory with PROM.EXE and PAL.EXE. To create the file simply create an ASCII string of device ID codes separated by a semicolon. The device ID codes can be found in the upper right hand box on the programming menu or you can print a list of devices with their associated ID codes by entering `<+>` or `<*>` in the vendor selection menu.

## CHECKSUM DISPLAY

A four digit (HEX) checksum will be displayed in the programming menu. With many of the larger memory devices, four digits is not enough to actually display the full calculation. The extended checksum up to 7 digits can be read with the `<X>` command in the main programming menu.

## MAIN PROGRAMMING MENU

Following the device selection, the main programming menu is displayed on the screen. This menu is composed of a list of commands. Command selection is performed with a single key stroke (the letter corresponding to the command in the menu) or with the cursor-keys and PRESS `<ENTER>`. Commands may be selected with either upper case or lower case letters: Optima DOS will understand both. The function of each command is described in the following paragraphs of this chapter. In the top right-hand corner of the screen the vendor, device and device ID code that have been chosen are displayed. If this does not correspond to the vendor or the device that is to be programmed, then the user may change the device type and the vendor by executing the command `<C>` (Change device type), see below.

## PROM PROGRAMMING COMMANDS

```

OPTIMA: Memory Programmer Bx/98
A = Set address range
C = Change device type
▶ C = Change device type
D = Execute DOS command
E = Edit pattern in memory
I = Input file from disk to memory
L = List contents of memory
M = Multiple device programming
N = Select multisyte unit
P = Program device
Q = Quit
R = Read device
S = Special device functions
U = Verify device to memory
W = Write HEX or BINARY disk file
Z = Device serialization options
F,H,O,X : additional commands
Dec 22 1998

Device code is 602076
AMD Am28F512
Device type selected

0000
Checksum

ALL BYTES
Data-Type

Enter command _

```

Figure 5.4: PROM Programming Menu

### <A> SET ADDRESS RANGE

Sometimes it is necessary to program a set of bytes in a PROM without programming the entire device. The set address range command **<A>** allows the user to work on and affect any portion of the entire address range of the device without influencing other areas. The user will be prompted for the start and end addresses of the range to be programmed. The PROM programming utility will check on the values entered to ensure that values entered do not correspond to an address range larger than the total address range of the device.

During blank check, program and verify, only the range selected will be addressed. The added versatility of being able to specify the start address of the device in memory to be anywhere within the full address space of the complete memory buffer allows smaller PROMs to be combined together into one larger PROM. See the 'Read' command for more details on the combine function. If a range other than that which is normal for the device is set, a box will appear on the screen to display the currently active range.

**Note:** The checksum displayed on the screen refers to the range selected.

### <B> BLANK CHECK

This command verifies that the device is blank. The device in the socket is read and the contents are compared to the expected 'erased' condition. The device type selected must match the physical device inserted in the socket. Some devices require special blank check sequences, Optima DOS automatically uses these sequences if required.

EEPROM and Flash EPROMs require no blank check before programming, they will be automatically erased as part of the programming cycle. The **<B>** command for these devices is not significant.

**<C> CHANGE DEVICE TYPE**

The active device type can be changed using this command. This command will take you back to the vendor selection menu described in the Device Selection section at the beginning of this chapter.

When the device type is changed, the active memory buffer area is automatically changed to accommodate larger or smaller devices as necessary; however, data previously stored in the memory buffer remain unchanged until either a device is read, or data is read from the disk.

**<D> DOS COMMAND**

This allows MSDOS commands to be executed without leaving the Optima DOS environment. Note that the MSDOS 'COMMAND.COM' file must be either in the SPRINT directory, or locatable via the DOS PATH command for this to work. If large EPROMs are select, there may not be enough room to load the MSDOS command file. Once loaded, any normal DOS command can be entered, in the same manner as in DOS itself.

Optima DOS will remain in the DOS command mode until 'EXIT' is typed on the keyboard. On completion of the DOS command, the user is returned to the main PROM utility menu.

**<E> EDIT PATTERN IN MEMORY**

```

                                HEX editor
AMD Am28F512                    Type of data: GANG / ALL    BYTES
 0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
000000: ff ff
000010: ff ff
000020: ff ff
000030: ff ff
000040: ff ff
000050: ff ff
000060: ff ff
000070: ff ff
000080: ff ff
000090: ff ff
0000a0: ff ff
0000b0: ff ff
0000c0: ff ff
0000d0: ff ff
0000e0: ff ff
0000f0: ff ff

<ESC> exits, HEX data is stored, ' or " enters ASCII mode, F1 for Help

```

**Figure 5.5:** Viewing the contents of the memory buffer

Selection of this command allows the user to examine and modify the contents of the memory buffer. This Screen oriented editor displays 256 bytes (100 HEX) from the buffer at any one time. Data is displayed in HEX format. Changes from the keyboard may be input in HEX or ASCII format.

After typing <E> for Edit, the screen will display a 256 byte page of data from the memory buffer starting at the address 0, (unless a different start address has been defined using the A command). The cursor control keys may be used to point to any address location in the memory.

<b>&lt;home&gt;</b>	go to address 0000
<b>&lt;end&gt;</b>	go to the last address of the device
<b>&lt;pg dn&gt;</b>	move down 256 bytes
<b>&lt;pg up&gt;</b>	move up 256 bytes

The Help screen lists extra data control functions in addition to the cursor movement keys. Help is displayed with the <F1> key. The data control functions listed in the help screen are:

<F1>	HELP
<F2>	GOTO
<F3>	COPY
<F4>	SWAP even odd
<F5>	FILL
<F6>	SEARCH
<F7>	REPEAT SEARCH
<Ins>	Toggle insert mode

In order to enter ASCII TEXT mode, the user may type the apostrophe <'> key. This allows the entry of ASCII text data directly. <ESC> returns from TEXT mode to HEX mode.

Data may be entered while in HEX mode by simply typing in the hex data. The ASCII equivalent will be displayed at the end of each line.

The <F2> key is used to directly jump to any location in memory (GOTO). After pressing <F2>, the user will be prompted to enter the destination address.

The <F3> key is used to copy blocks of memory. After pressing <F3>, the user will be prompted for the start address, number of bytes (in Hex) and destination address.

The <F4> key will swap all even and odd bytes in memory.

The <F5> key is used to fill portions of memory with any data pattern, the user will be prompted to enter the start address, end address and fill data.

The <F6> key allows a search in memory for a set of HEX bytes. Enter after the prompt an even number of digits, 2 digits for each byte, after <Enter> is typed, the search will start. For example, if the sequence in memory you would like to find is:

12 34 06

then enter 123406..

The <F7> key will repeat the previous search..

The **Insert** key will enable the insert mode. Normally data entered overwrites data in memory. With the insert mode enabled, entering 1 or 2 hex digits will cause all locations after the cursor location to be copied up one byte.

The <.> key or the <ESC> key will exit the editor, with a new checksum being calculated, and a return to the main menu. For very large devices, on slow computers, this can cause some seconds of delay. The correct checksum is displayed in a box to the right of the main menu screen.

```

                                HEX editor
Intel 87C51                      Type of data: GANG / ALL   BYTES
  0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
Security byte: 00 unprotected, 01-write 03-read protection
001000: 00

<ESC> exits, HEX data is stored, ' or " enters ASCII mode, F1 for Help

```

**Figure 5.6:** Use of control bytes / Encryption array

Some devices have additional control bytes after the actual data area. For example the 87C51 device shown in the example has data from 0000 to FFF and a control byte at 1000. These special bytes are explained with clear text in the editor. Simply type **<end>** on the keypad to access these bytes. They may then be edited like any other data in memory. Same for the Encryption array (1001 to 101F). See the section in this chapter called Device Specific Information for detailed information.

#### **<F> FILL MEMORY**

A range of buffer memory can be initialized to any value with this command. After **<F>** is entered, the user will be prompted for the hex start address, end address, and one byte of data. The addresses and data are separated by commas. After pressing **<Enter>**, the range of memory will be initialized to the value entered. Fill will always change at least one byte on execution.

**<I> INPUT FILE FROM DISK TO MEMORY**

```

Enter filename to read: _
Drive C:  Directory \EXPERT.A94\EXAMPLES          File type *.*
  --
24U10.PLD      18U10.PLD      20R410.PLD      22U10.PLD
7C331.PLD      26CU12.PLD     26U12.PLD       3T08DMUX.PLD
E16P4.PLD      7C332.PLD     DEMO.COM         DEMO.HEX
MAX20.POF      GAL6001.JED    GAL6001.PLD     INTEL.HEX
MAX7032.POF    MAX28.POF      MAX44.POF       MAX68.POF
PLX448.PLD     MAX84.POF      MOTOROLA.HEX    PLS153.PLD
TEP900.PLD     T16R4.PLD     T18N8.PLD       TEP600U.PLD
                TS26CU12.PLD   TST16R4.JED     VECTOR.PLD

F1: HELP
or use cursor keys to highlight item, then press Enter to select

```

**Figure 5.7:** Loading a data file from disk

The **<I>** command is used to load the data from the file selected into the memory buffer. The system will automatically default to the last drive, directory and file type (extension) that was used. All files with the designated extension will be displayed. You may type in the file you want to input or use the arrow / TAB key to select a file or change any of the defaults.

For more details refer to the File Input Mode in the introduction section of this manual.

After a filename is selected, the user will be prompted to enter various options about the file. The keyboard cursor up and down keys are used to point to the option required, and the value can be typed-in. The file type, Binary or Hex may be selected (default is Binary). The byte/word extraction can be specified (default is All). The starting address of the data in the file on disk can be entered, and the starting address of the data in the buffer memory can be entered (default is the 'bottom' address defined by the **<A>** address range command). A return without changing any parameters starts the load process. The 'Auto' option default for the start address allows Optima DOS to find the most likely starting address for Intel HEX files with physical starting addresses in segments other than segment 0. Inserting a 0 in this field will force the physical address 0.

```

File input mode

File to Read : EXPERT.A94\EXAMPLES\DEMO.COM
Select binary/other file format(B,H,M,L) : B
Select Byte: All/Even/Odd      (A/E/O)
          Byte: 0/1/2/3 of four  (0/1/2/3)
          Upper/Lower Word of 32-Bit (U/L)
Merge 8-bit files -> 16bit (M even,N odd) :A

Enter start address of data in file      : auto :
Enter start address of data in memory    : auto :

Enter return to start, ESC to exit, up or down cursor to change options

```

**Figure 5.8:** File Input Mode

There are 9 different input options for the data:

<u>command</u>	<u>function</u>	<u>file data</u> <u>size</u>	<u>EPROM</u> <u>size</u>
A	All for 8 bit data	any	same
E	Even for even bytes of 16 bit data	16	8
O	Odd for odd bytes of 16 bit data	16	8
0	Byte 0 of 32 bit data	32	8
1	Byte 1 of 32 bit data	32	8
2	Byte 2 of 32 bit data	32	8
3	Byte 3 of 32 bit data	32	8
U	The last 2 bytes of 32 bit data	32	16
L	The first 2 bytes of 32 bit data	32	16

The user can select any one of these options, or accept the default (A).

Before any data is read-in, if the start address is equal to 0000, the memory buffer will be reset to either 00 or FF, depending on the buffer default option. See the 'O' command for more detail. Any extra bytes (security, initialize, reset control etc) will be set to 00. Note that these extra bytes are located after the actual memory data.

If the start address is not equal to 0000, the existing data in the memory buffer is not changed. This is done to allow files to be merged and then be programmed into devices.

Data will then be read from the specified file using the options selected, until either the top address is reached, or the end of the file is reached. The number of bytes read from the file will be displayed on the screen. After reading, a new checksum value will be calculated. For very large devices, this may cause a few seconds of delay. The correct checksum is displayed on the main menu screen.

If the EPROM memory size is larger than available memory, the excess data will be read in, formatted and stored in a temporary file (DAT.CNF) see the Option menu for details.

## INPUT FILE FORMATS

Twelve different file types are supported,

### a) Binary Format

If this format is selected, the data in the file will be read until either the end of file or the top address of the PROM is reached. A non-zero start address offset in the data file on the disk may be specified, the default is zero. Finally a destination address offset for the data in the memory buffer can be selected. The default for the destination address offset is the beginning of the programming range (see Set Range command **<A>**).

### b) HEX Space

This is a string of bytes in hex format, separated by a space character. Returns and line feeds are ignored. An example of the format is shown below, it is also on the distribution floppy disk (file "DEMO.HEX"). HEX space files are terminated by an ASCII ETX (HEX 04) character.

```
STX 12 23 34 45 56 67 78 89 90 01 12 23 34 45 56 67 78 89 12 23 34 45 56 67 78 89 90 ETX
```

### c) HEX with checksums

The following standard HEX file formats are supported:

**Intel HEX, standard, extended and 32 bit  
Tektronix HEX, standard and extended  
Motorola S Record, 16, 24 and 32 address modes.**

These formats are supported according to each manufacturer's definition. A start address of the data in the file may be selected. With Intel and Tektronix HEX formats, both 16 and 20 bit addresses are supported, with Motorola S Records, 16, 24 and 32 bit addresses are supported.

### d) 286 Object module format.

Only boot loadable files are supported, selected with **<H>** like the HEX file format.

### e) MOS technology file format.

It is selected in the File input mode menu with the **<M>** instead of **<B>** for binary.

**Optima DOS** automatically determines if the HEX file is SPACE, Intel, Tek or Motorola.

## <L> LIST MEMORY CONTENTS

This command displays in tabular form the contents of the memory buffer. When **<L>** for List is entered, the user is prompted for the start and end addresses. A comma separates these addresses. If the user wishes the display to go to the printer, a ', \* ' is typed after the end address. Memory is displayed in HEX, 16 (HEX 10) bytes per line, from the start of the address range to the end of the address range. A pause is inserted after each full screen to allow the user to examine the contents of the memory buffer. The next screen of data can be displayed by entering a **<Y>** or return at the end of each page. The continuation of memory display may be terminated by typing **<N>** or **<. >** at the end of the currently display page. The ASCII equivalent of the HEX data is also displayed at the end of each line, for data values between 32 and 127 decimal.

**Note:** (For OPTIMA / PLUS48 only) If you have a problem printing, you may need to move your printer to LPT1 (address 378) and put the programmer on LPT2 or LPT3.

### **<M> MULTIPLE DEVICE PROGRAMMING**

This command sets up a counter for programming a fixed number of devices. After entering the number of devices to program, the operator need only insert the device, and type any key to program. Programming stops after the unit count is reached. Only successful programming operations will increment the counter.

### **<N> SELECT MULTISYTE UNIT (appears only if a Multisyte programmer is installed! )**

If a Multisyte system is installed, then the 'N' command can be used to :

- a) Select a different TOP, numbered 1 to 8, left to right. The device menu will appear with the device list that matches the installed TOP. The **<SPACE>** key can be used to return to the main menu without changing the device selection. Note that the device lists vary according to the installed TOP.
- b) Select all TOPs in parallel that have the same type as the previous selected. if the response to the 'N' command is a '0', then all units will be selected. This option allows from 2 to 8 devices to be programmed at the same time - with no difference in the programming cycle time. Each device has its own complete set of pin drivers.

The Multisyte GANG mode increments the COPY counter by the number of units programmed. Importantly, GANG mode handles all error conditions in case a device fails in any operation. Any sockets that are empty during the programming will be ignored. Depending on device type and the operation executed, sockets containing devices with errors will be disabled for the rest of the operation, and indicated with a red LED. The verify and blank check operations will execute on all devices at once, if the devices do not have the same contents, then the cycle will repeat automatically to identify the defective devices, marking the devices with a RED or GREEN LED.

Read Device uses unit #1, otherwise all operations occur on all devices in parallel, empty sockets are skipped.

**<O> DISPLAY PROM OPTIONS**

```

PROM variables                                Intel 87C51
Device ID check                               0n
Blank check                                  Enable
Set Pause delay to                           11614
Set Read Back delay to                       2
Continuity Check                             Enable
Vpp voltage is                               12700 mV
Vcc during programming                       5000 mV
Device type/size (bytes)                     Microcontroller / 4096
Vendor ID / Device ID is                    not used
Initial pulse width                          100 us
Maximum pulse count                          25

Programming algorithm                         QuickPulse

Number of feature bytes                      1
Vcc during Verify                           5000 mV
Screen display mode                          fast-color
Default Buffer contents                      FF Hex
Beep on programming done                    ON
disk for DAT.CNF                             C
swapping pagesize(128/256kB):               128k
Enter return to quit, ↑ or ↓ to move cursor, space+data to change options

```

**Figure 5.9:** The PROM Option Screen

This command will display on the screen, the programming parameters for the device selected. It can be used to compare the voltages selected by **Optima DOS** with the voltage recommended in the data sheet of the device.

All other parameters displayed can be useful information about the device selected, and the operation mode of **Optima DOS**. The user can change any parameter shown (**at his own risk!!**). These parameters will not be saved once the Optima DOS is exited (except screen display mode and the DAT.CNF location and size).

**Note:** The 4 delay options (Optima / PLUS48 only) are only used in the Data I/O factory.

**DISKSWAPPING**

Device data files are loaded into PC RAM prior to programming. Because Optima DOS is sharing the 640K RAM allocation with DOS, there is not enough space to contain the entire data file for large EPROMs. To compensate, the data is stored in a temporary dat.cnf file and loaded into RAM in blocks. The default is 128K bytes but can be optionally set to 256kB.

You may select the disk to use for the swap file. Note that there is very little performance increase if a RAMDISK is used instead of a physical disk for this function. Disk cache applications like Microsoft SMARTDRIVE work, and can increase performance.

**<P> PROGRAM DEVICE**

When **<P>** for Program is entered, the device in the socket can be programmed using the data stored in the memory buffer. The device type selected by the user must match the physical device in the socket. The user is prompted to confirm the programming of the device by typing a **<Y>**, any other key will abort the command. This has been implemented so that accidental programming of devices may be avoided.

After confirmation that programming of the device is to be performed, non-erasable devices will be checked for erasure (Blank check). If the device is not blank, **Optima DOS** will test if the pattern in memory can be overprogrammed on top of the existing pattern in the device. If so, the user will be prompted to

'overprogram'. Entering <Y> will allow **Optima DOS** to program on top of the existing pattern. Erasable devices will be erased automatically before programming.

The device will be programmed using the algorithm defined by the manufacturer. Intelligent, Flashrite, Rapid, QuickPulse, Quickpro, Page Mode and all other fast programming methods are used to keep the programming time to a minimum. The programming algorithm used is automatically determined by the vendor/device type chosen.

For devices with long programming times, the number of 512 byte blocks remaining to be programmed is displayed to indicate that the programming operation is processing. This is useful in large devices not using QuickPulse type algorithms. This 'ACTIVITY' display is not shown when QuickPulse devices are programmed. When the 'ACTIVITY' display is shown, programming can be interrupted with the <ESC> key.

After programming, the contents of the device will be verified against the contents of the memory buffer to confirm the successful programming of the device. Following a correct verify operation, when everything has been successfully completed, the message **PASS** appears on the screen.

If there are any errors, they will be shown on the screen, and the blinking message **FAIL** will appear. Otherwise the message **PASS** will be displayed. The number of devices successfully programmed with the same contents will be displayed in a box to the right of the PASS/FAIL box.

#### <Q> QUIT

Returns the user to the **Optima DOS** opening menu, or to DOS if PROM was started from DOS.

#### <R> READ DEVICE

This command reads the contents of the device in the socket into the memory buffer. Note that after data has been read into memory, the device type may be changed to the same size or smaller without altering any data stored.

When registered PROMs are selected, the Reset Byte and Programmable Synchronous Enables (if any) are automatically read into the buffer memory for correct duplication or storage.

After reading in the data, a new checksum value will be calculated. For very large devices, this can cause some seconds of delay. The correct checksum is displayed on the main menu screen.

With the <A> command (= set address range), an address range larger than the physical device may be emulated. This allows the user to read in several small EPROMs, at successively higher contiguous memory addresses in the memory buffer. At the same time, this feature enables the user to combine the data contained in a number of small EPROMs and to program this data into one larger EPROM. The editing buffer is 32768 bytes (32 Kbytes) in size. An EPROM in the socket can be read into any portion of this editing buffer. When a larger device is selected with the <C> change device command, this editing buffer is not cleared.

Example: To combine the contents of four 2764's and to program this data into a single 27256 the following procedure may be performed.

- a) Select the vendor and device for the 2764 devices that you wish to read.
- b) Select the address range 0 - 1fff (A command)
- c) Read in the lowest addressed 2764 (R command)
- d) Select the address range 2000 - 3fff (A command)
- e) Read in the 2764 with the next address (R command)
- f) Select the address range 4000 - 5fff (A command)
- g) Read in the 2764 with the next address (R command)
- h) Select the address range 6000 - 7fff (A command)
- i) Read in the highest addressed 2764 (R command)
- j) Change the device type to 27256 (C command)
- k) Insert the 27256 device into the socket and program it (P command)

**<T> GANG/SET PROGRAMMING OPTIONS ( appears only if TOP432 is installed)**

**G)** GANG mode, all four sockets of TOP432 are programmed with the same data.

**E)** Eight bit set mode, places sequential data into each EPROM 'stacking' the EPROMs together to hold the entire data loaded into memory.

**S)** Sixteen bit set mode, puts all even data into the sockets 1 and 3, all odd data into the sockets 2 and 4. Two sets of 16 bit wide data can be programmed at a time.

**T)** Thirty two bit set mode, puts the 0th, 4th, 8th byte (etc) into socket 1, the 1st, 5th 9th byte (etc) into socket 2, and so on

**Note: When in any of the three SET-modes, the data in memory is arranged in the following way:  
 Byte 0 - socket 1 , byte 1 - socket 2 , byte 2 - socket 3 , byte 3 - socket 4  
 byte 4 - socket 1 , byte 5 - socket 2 , byte 6 - socket 3 , .....**

**The difference between the different SET-modes is only in the file-input. When changing SET-modes, the data remains unchanged in memory, therefore always first select SET-mode and then input your data from file.**

**<V> VERIFY DEVICE**

Compares the contents of the device in the socket to the data in memory buffer. Any differences are displayed in clear text, with memory data, device data and address. All features of the device are verified, including extra reset and enable bits and architecture bits.

**<W> WRITE BUFFER DATA TO DISK**

Writes the data in the memory buffer onto the disk using the filename supplied. After entering **<W>**, the user will be prompted to enter a MSDOS style path. A drive label or path may optionally be entered. The default path is accepted if **<Enter>** is pressed without any other characters. The filename may then be entered.

The user may select either Binary **<B>** or Intel Hex **<H>** format to write the data.

When writing the buffer to disk, the number of bytes written corresponds to the size of the currently selected address range.

In case a file with the same name exists, the user will be prompted to confirm erasure of the previous file.

**<X> EXTENDED CHECKSUM CALCULATION**

A four digit (HEX) checksum will be displayed in the programming menu. With many of the larger memory devices, four digits is not enough to actually display the full calculation. The extended checksum up to 7 digits can be read with the X-command. To activate the extended checksum enter **<X>**, then 5,6 or 7 to select the length, and the checksum will be displayed.

**<Z> DEVICE SERIALIZATION OPTIONS**

This command writes a serial number to the device, which is incremented with every device programmed. In the file-input menu you designate a file to store the serial number and any options. Use the arrow keys to select the file you want, or type the complete path and file name, then press **<Enter>**. The Device Serialization Options screen is displayed.

```

Device serialization options
S/N Options file:  C:\OPTIMA\NEU.SN
Next S/N:         0 dec.
S/N stored in:    HEX
S/N Digits:       1
S/N Start Addr.:  0 hex.
Byte Order:       Big Endian
Store S/N bytes:  NO S/N
press F1 for help

```

If you want to create a new file, make sure that you add the extension, **.SN**. If you press **<ESC>** instead of **<Enter>**, you will exit from the input menu, nothing will be saved, and serialization will be disabled.

Programming a serial number into devices requires the following parameters. You must enter some value for each parameter:

- **Next S/N**—the serial number that you want to be written to the device in the next programming cycle. This number must be in decimal base (for example, **0\_dec.**). The Next S/N will be written to the buffer before each programming cycle.
- **S/N stored in**—the numerical format for the serialization. This can be either ASCII (each digit from 30 hex. to 39 hex.) or HEX (each digit from 0 hex. to A hex.).
- **S/N Digits**—the number of digits to use in the serialization (8 maximum for HEX, 9 maximum for ASCII). This allows up to 999,999,999 units with ASCII serial numbers, and 4,294,967,296 units with HEX serial numbers.
- **S/N Start Address**—the address in the device of the serial number's first byte (Start Address in hexadecimal mode). All of the serial number must fit into the device, or serialization cannot take place.
- **Load S/N bytes**—identifies bytes written to the buffer, and their order as follows:
  - **All**—all the bytes are written
  - **Even**—only the even bytes in the serial number are written. If this option is selected, the S/N Digits number must be even.
  - **Odd**—only the odd bytes in the serial number are written. If this option is selected, the S/N Digits number must be odd.
  - **0 of four, 1 of four, 2 of four, 3 of four**—only the bytes in the corresponding positions multiple of four in the S/N are written. If this option is selected, the S/N Digits number must be 4 or 8 (only 8 if stored in HEX)
- **NO S/N**—disables serialization. The Serialization option can be disabled at any time by pressing **<ESC>**.

## PROM ERROR MESSAGES

The following is a list of common error messages for the PROM utility.

**1) Disk Full, no data written.**

The disk drive used for the write operation was too full to contain the entire file. The partially written file has been erased.

**2) PROM not blank**

The PROM contains data. The data in the **Optima DOS** memory buffer cannot be over-programmed into the device.

**3) O.K. to program on top of existing pattern?**

PROM was not blank, the new pattern can be over-programmed onto the existing pattern. Enter 'Y' to commence programming.

**4) Manufacturer xx Device yy not recognized**

Auto ID could not identify the device in the socket. Either the device does not support silicon signature, or the ID does not match the device selected.

**5) Error with 'auto' in start address of data in file at: xy**

If the start address of the file is set to 'auto', and the first address in the HEX file is higher than the size of the selected device, then the first address in the HEX file sets the automatic start address. All data from the HEX file are shifted down to 0, by subtracting this start address.

If a later address is lower than the first address read, the upper message will appear. The data you have read is probably not correct. Please read the file again with the correct start address of the data in the file. It is shown on screen.

## 6) Checksum Error occurred during read operation at address xy

In the INTEL HEX file format the checksum of every record is tested. An error message will be shown on screen, but loading the file is continued.

## DEVICE SPECIFIC INFORMATION

Extra PROM and microcontroller features like Reset, Synchronous Enable, Security Locations and Encryption array are automatically programmed in devices which have them; these features are controlled by extra data bytes. See the PROM notes section for specific details and examples. The extra bytes in the memory buffer may be set using the buffer edit utility (see above), and can be stored to the disk with the **<W>** command.

1. Registered PROMs that have a programmable initialization byte, have this byte stored after the last data byte. The location in memory of this byte is demonstrated by these examples:

Device size	Type	initialize byte location (HEX)
1024	7c235, 27S35, 27S37	400
2048	7c245, 27S45, 27S47	800
8192	7c268, 7C269, 27C265	2000
32768	27C203	8000
65536	7C287	10000

**Optima DOS** automatically programs this data into the PROM during the programming cycle. The byte can be read in from a device or from a disk data file. When reading a device, **Optima DOS** automatically determines the reset byte value.

2. Proms that have additional programmable features have the data stored in a control byte after the reset byte value. For example :

Device size	Type	Control byte location (HEX)
2048	7c245, 27S45	801
8192	7c268, 7c269, 7c265	2001
32768	7C277, 7C279	8001
32768	27C203	8001

**Optima DOS** automatically programs this data into the PROM during the programming cycle. The byte can be read in from a device or from a disk data file. When reading a device, **Optima DOS** automatically determines the control byte value.

3. Microcontroller programming is identical to PROM programming, except that an additional byte contains a flag for the security fuse. This byte is located after the last data byte. If a device contains multiple fuses, the first fuse will be bit 0, the second bit 1, etc.

GI-Microchip PIC16Cxx family. This family has five special bytes after the data-area. The first four bytes are the customer ID code. Notice that only the lower Tetrade can be accessed. The last byte is the configuration byte. The bits have following meaning:

Bit0 and Bit1 set the oscillator type	(default RC oscillator)
Bit2 = 1 Watchdog Timer Enabled	(default)
Bit3 = 1 Memory unprotected	(default)

Some microcontrollers have a special Encryption array. This array is located after the security byte. This array can be used to encrypt the data in the device. This can be useful if you do not want to secure the device, but avoid that someone else see the original data.

4. Most newer EEPROMs have a Software data protection feature. If we support this feature with the device, an additional byte is at the end. To enable after programming the protection write 01 in this last byte. (i.e. : 28C256 address 8000H, data 01)



## 6. Technical Specifications

**Size.** The PLUS48 is smaller than 22 x 18 x 8 cm. This package incorporates the printer port interface, 48 pin drivers, 16 high speed clock drivers and relays for 27 power and ground pins.

**I/O address.** Uses one of the port addresses 278, 378 or 3BC hex. These are IBM PC compatible printer ports with 8 data lines, 4 control lines, 5 status lines and ground. DOS operated computers offer these ports as LPT1, LPT2 or LPT3 under the above addresses depending on the hardware used. Optima DOS will automatically search for the correct address.

**Power.** Power is supplied through an external switching power supply suitable for a wide range of VAC inputs. The power supply is rated for UL, CSA, and VDE safety standards.

**Socket.** The unit includes a 48 pin DIL socket that accepts 8 to 48 pin packages in 300 and 600 mill widths. Optional adapters that plug into the 48 pin DIL adapter are available for non-DIP packages and devices larger than 48 pins.

**Pin drivers.** All 48 pins are capable of the following:

Output:	
Logic 0	4 ma drive to ground
Logic 1	Voltage pull-up via 10K resistor
Vpp1	200 ma per pin, 900 ma total
Vpp2	200 ma per pin, 900 ma total
Device Ground	Relay are used exclusively for device grounds
Vcc	Relay supplied Vcc in the range of 3 to 9 volts continuous, with peak programming pulses of 25 volts/500 ma
Vih	Device input voltage programmable from 2.0 to 12 volts
Fast Clock	12 fix wired and 3 variable 3 ns rise time clock pins
Oscillators	2 fix wired and 2 variable 5 Mhz oscillators for programming 87C51 and similar devices
Noise reduction	Dynamically switched bypass capacitors on all pins

(Pulse currents up to 800 ma for under 100 us are supported)

Input:

CMOS input, 1.5 volt threshold, protected up to 2000 volts ESD, 30 volts continuous

**Memory.** The system must have free memory available to store the Optima DOS and device data. PC must have 640K RAM installed.

## OPTITEST SELF-TEST Software

```

Checking Printer Port
Please remove device or adapter from the socket ... then press any key_

```

Starting Self-test, either from the Optima DOS main menu (#5) or from DOS results in the above screen being shown. Normally, the user should check that the socket is free of any devices, and then the SPACE key can be pressed to start the test. The test will check all important parts of the programmer and of programmer to computer interface.

```

OPTIMA Self Test      Bx/98
socket type: TOP48      syte 2
use OPTITEST -C option to calibrate voltages
OPTIMA Copyright (c) 1991 - 1998

Testing all pins, SA0, SA1, Shorts
Testing Upp driver function, I/O cycle test
Testing relays and clock drivers
Check LED COLOR:
Check LED OFF
register check
Testing internal logic 012345
Testing RAM memory
Testing programmer bus
completed: 100%
Testing ASIC input voltage and Diode/Resistor connections
Test High Voltage drivers, floating turnoff Pass 3_
All Tests ok

Enter any key to exit

```

LEVEL  
ALL

Serial #  
20151

ioadd  
378s

system type  
Octal

SMP expires after  
C/98

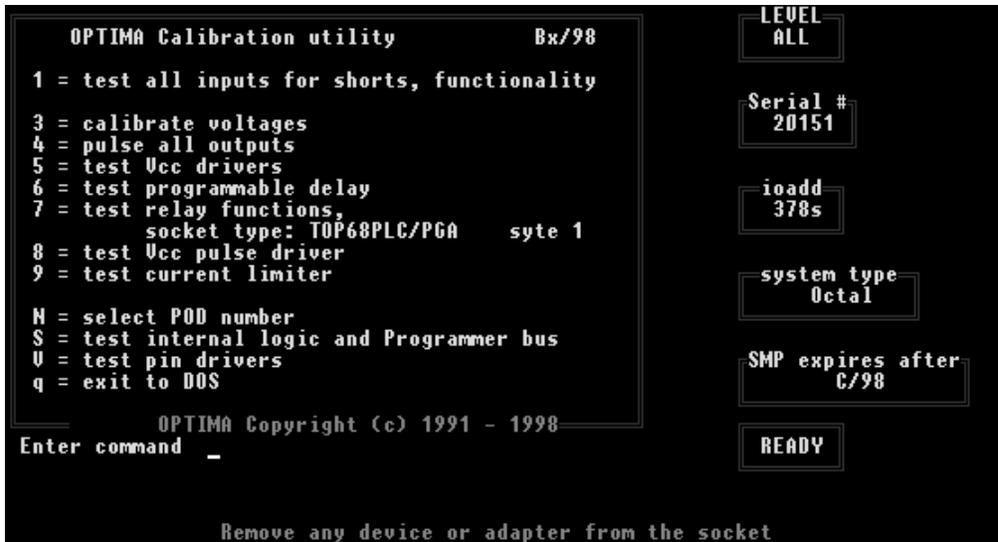
PASS

The status box in the lower right corner shows **PASS** indicating all is functioning OK.

Note that the serial number and SMP expiration date are shown for easy reference. SMP is the Software Maintenance Plan that keeps you up to date with the latest algorithms. Updates are free of charge until the update shown. There are three updates per year, #1, #2 and #3. They are label A, B and C, and are typically available in May, September and December.

If your programmer is configured for the entry level (32 pin support) then the status boxes in the upper right will show PROM (32 pin) and PAL (28 pin). Call SMS for an upgrade to unlimited pin number support.

The printer port address used is also displayed for reference.



If <ESC> is entered, then the individual tests and calibration of OPTITEST are shown. For each of these tests, an oscilloscope and/or voltmeter will be required. Simply follow the on-screen instructions for each of these tests. If Optima DOS should fail any of these tests, we will ask you to return the unit to us for repair, because of the SMD technology used, there are no user replaceable components inside.

Normally the only user options will be to calibrate the unit, once per year.

**SOCKET PINOUT**

Pin	Ground Relay	Vcc Relay	Vpp1	Vpp2	Fast Clock	Oscillator
1			Y	Y		
2			Y	Y		
3			Y	Y		
4			Y	Y		
5			Y	Y	Y	
6			Y	Y		JP11
7			Y	Y		
8			Y	Y		
9			Y	Y		
10			Y	Y		
11			Y	Y	Y	
12		X	Y	Y	Y	
13			Y	Y	Y	
14	JP2	X	Y	Y	Y	
15	X		Y	Y	Y	
16			Y	Y	Y	
17		X	Y	Y	Y	
18	X	X	Y	Y		
19		X	Y	Y		
20	JP3		Y	Y		
Pin	Ground Relay	Vcc Relay	Vpp1	Vpp2	Fast Clock	Oscillator
21			Y	Y		
22			Y	Y	Y	JP10
23			Y	Y	Y	X
24	X		Y	Y		
25		X	Y	Y	Y	
26			Y	Y		
27			Y	Y		
28		X	Y	Y		

29	X	JP7	Y	Y	
30	X	X	Y	Y	
31	X	X	Y	Y	
32	X	X	Y	Y	
33	X	JP6	Y	Y	
34	X	X	Y	Y	
35			Y	Y	Y
36		X	Y	Y	
37		JP5	Y	Y	
38		X	Y	Y	
39			Y	Y	
40		X	Y	Y	
41			Y	Y	
42			Y	Y	
43			Y	Y	
44		X	Y	Y	
45			Y	Y	
46			Y	Y	
47			Y	Y	
48			Y	Y	
spare S7	JP8	JP9			
spare S6		X			
spare S5	X				
spare S4					X
spare S3					X
spare S2					X
spare S1					X
spare S1					X

Spare connections S1 to S7 allow future devices with ground, power or clock requirements that are unknown today. If required, instructions for their use will be provided later.

## 7. Error Messages

### ERROR RESPONSES

Backward devices or poor socket contact will result in a **pin continuity error message**. The user will be prompted to display the pins that failed. By entering <Y> for yes, the failing pin can be displayed on the screen. The pins in parentheses (**xx**) are the expected ground pins of the device. If the ground contact fails, all pins including (**xx**) will be shown. The reversed device or bad pins can be examined at this point. Note: not all devices permit continuity testing on all pins, Optima DOS automatically excludes those pins that are not testable. In some cases, the engineer may wish to skip the test and continue programming. This is not recommended, but the special key <#> will allow the user to do so. Normally the operator will type <n> to return to the main menu before correcting the device condition.. Please note that if the error is ignored, damage to the device could occur.

Device short test will determine if any pins of the installed device are shorted. **Pin XX shorted, continue?** is the error message, always enter <n> since this test provides added protection to the programmer hardware, and a defective device could under certain cases damage the programmer. Please remove and discard the device.

Offset test counts the number of pins in the package, and rejects devices that do have the **WRONG PIN COUNT**. The error condition cannot be skipped.

Optima DOS checks its own pin drivers before every programming cycle. **VPP PIN XX FAILED** is the message that is displayed. The message is fatal, please run the Optima DOS self-test to determine the exact failure and contact your Optima DOS distributor.

Relays are mechanical, and Optima DOS checks the contacts of the relays for stuck open or closed conditions. Should a relay fail, the user will be asked to run the Optima DOS self-test to make a detailed report of the problem.

Assuming all hardware is OK, then the device is check for proper silicon ID. **Incorrect Silicon ID** will result if this happens. There is no option to skip this message. One reason why this error appears is, if you try to program a new version of a device and Optima DOS is not updated, please contact your local Optima DOS distributor for a SOFTWARE MAINTENANCE PLAN - SMP.

## PLD Error Messages

This is a summary of common error messages and how they should be managed.

### 1) **Error: display detail report? <Y/N>**

This message is output by blank check or verify. It indicates that the data read from the part did not match the expected data. This means that the device was not empty (blank check) or is not programmed correctly (verify).

### 2) **Skip blank check? <N>**

If a device was not blank during a programming cycle, it is possible to continue programming by entering <Y>. If you get this message during blank check, it means that Optima DOS has compared the device to the contents and RAM and determined that it is possible to program the new pattern into the device.

### 3) **Programming device - operation failed**

The device could not be programmed. Present silicon technology allows a 99% or better average programming yield. Depending on the manufacturing lot, the yield you experience may vary. A common cause of programming failure is with erasable devices, which have been damaged in the application. If a device is removed from a PC board, erased, and then programmed, there should be no problem. However, if the device was damaged in the PCB, programming will fail. If the device is UV erasable, be sure that the device was exposed long enough to assure erasure.

### 4) **Manufacturers Electronic Signature incorrect, Silicon ID error**

Silicon Signature allows Optima DOS to adapt the programming voltages and pulse widths automatically to a code provided by the vendor. The vendor provides specifications to Optima DOS before the silicon is sold to customers and we include these codes into the software. There are two possible causes for this message:

- a) The device has been damaged and is no longer programmable. Some CMOS devices are sensitive to latch-up and static discharge, this is a major cause of failures in CMOS products.
- b) The device is a new revision not supported by your version of Optima DOS. Contact your Optima distributor for an update.

### 5) **Preload not supported**

The device type selected does not support preload of test vectors.

### 6) **Test Vectors not supported**

This message occurs when using an adapter is used in the 48 pin DIP socket and the pin count of the device is greater than 48 pins..

### 7) **Continuity error**

The programmer could not detect a good electrical connection with one or more leads of the device in the socket.

## PROM Error Messages

The following is a list of common error messages for the PROM utility.

**1) Disk Full, no data written.**

The disk drive used for the write operation was too full to contain the entire file. The partially written file has been erased.

**2) PROM not blank**

The PROM contains data. The data in the **Optima DOS** memory buffer cannot be over-programmed into the device.

**3) O.K. to program on top of existing pattern?**

PROM was not blank, the new pattern can be over-programmed onto the existing pattern. Enter 'Y' to commence programming.

**4) Manufacturer xx Device yy not recognized**

Auto ID could not identify the device in the socket. Either the device does not support silicon signature, or the ID does not match the device selected.

**5) Error with 'auto' in start address of data in file at: xy**

If the start address of the file is set to 'auto', and the first address in the HEX file is higher than the size of the selected device, then the first address in the HEX file sets the automatic start address. All data from the HEX file are shifted down to 0, by subtracting this start address.

If a later address is lower than the first address read, the upper message will appear. The data you have read is probably not correct. Please read the file again with the correct start address of the data in the file. It is shown on screen.

**6) Checksum Error occurred during read operation at address xy**

In the INTEL HEX file format the checksum of every record is tested. An error message will be shown on screen, but loading the file is continued.



## Index

- Adapter, 1-10
- Address range, setting, 5-5
- Attention message, 1-11
- Automatic device selection, 5-3
- Batch mode, 1-19, 1-20
- Blank-checking a device, 4-5
- Bold type in this manual, 1-3
- Booster module for communications port, 1-7, 1-8
- Booting the programmer, 1-4
- Buffer memory, 4-3, 5-8, 5-11
- Cable for communications, 1-2, 1-3, 1-8
- Calculating the checksum, 1-10, 4-13, 5-4, 5-10, 5-16
- Calibrating the programming system, 1-5, 1-20, 2-2, 6-2
- Capitalization in this manual, 1-3
- CD, Optima, 1-2, 1-4, 2-1
- Changing a device, 4-5
- Checking for a blank device, 4-5
- Checksum, calculating, 1-10, 4-13, 5-4, 5-10, 5-16
- COM ports, 1-7
- Command line options, 4-1, 5-1
- Commands for PLD programming, 4-5
- Commands for PROM programming, 5-5
- Commands, running, 1-19, 3-1, 4-1, 4-4, 4-6, 5-1
- Communicating with the programmer, 1-5, 1-20, 2-2, 6-2
- Communications booster module, 1-7, 1-8
- Communications cable, 1-2, 1-3, 1-8
- Comparing devices to memory, 4-17
- Components of the programming system, 1-2, 1-10
- Computer requirements, 1-2
- Configuring the programming system, 1-3, 1-4, 2-1, 2-2
- Connecting the power cord, 6-1
- Contents of the programming system, 1-2, 1-10
- Continuity, checking, 1-5, 1-20, 2-2, 6-2
- Control bytes and encryption array, editing, 5-8
- Controlling scroll, 1-19
- Conventions in this manual, 1-3
- Converting a device into a JEDEC file, 4-6, 4-18
- Creating installation disks for Optima Windows, 1-4
- Cross-programming a device, 4-6, 4-18
- Data file, loading, 1-11, 4-13, 5-9
- Detecting JEDEC files, 1-19
- Device integrity testing, 1-20
- Device list, 1-9, 4-3, 4-4
- Devices that are supported, 1-2
- Devices, programming multiple, 4-14
- DIL socket, 6-1
- Directories, file, 1-18
- Disk drive, 1-4, 1-18, 2-1, 2-2
- Disks, creating, 1-4
- Diskswapping, 5-13
- DOS, jumping to, 4-6
- Drivers, 1-2, 4-11, 6-1
- Editing control bytes and encryption array, 5-8
- Editing memory contents, 4-7
- Editing test vectors, 4-8, 4-9
- Encryption array and control bytes, editing, 5-8
- Ending the task, Otestw, 1-5
- Error messages, 1-11, 1-21, 4-4, 4-21, 7-1
- Executing a DOS command, 4-6
- Exercising a device, 4-8, 4-9
- Expert/Multisyte option, 1-4, 2-2, 4-5
- FAIL indicator, 5-14
- Family of devices, 1-9, 4-2, 5-2
- Features of the programming system, 1-2
- File extension, 1-11, 1-18
- File folders, 1-18
- File formats that are supported, 5-11
- File standards, 4-20
- File, selecting, 1-18, 3-3, 4-13, 5-9, 5-10, 5-11
- Filling the buffer memory, 5-8
- Folders, file, 1-18
- Formats, file, 5-11
- Formatted text, 1-3
- Found slow printer port message, 1-7
- Gang programming, 1-14, 1-16, 4-14, 5-12, 5-15
- Hardware, installing, 1-3
- High-volume production, 1-8
- Initializing the programming system, 1-5, 1-20, 2-2, 6-2
- Inputting a file from disk into memory, 1-18, 3-3, 4-13, 5-9, 5-10, 5-11
- Installation disks, creating, 1-4
- Installing, 1-3, 1-4, 2-1, 2-2

- Integrity of a device, testing, 1-20
- Jack, power, 1-3
- JEDEC file, writing data to, 4-17
- JEDEC levels, 4-10
- JEDEC scanning, 1-19
- JEDEC, testing device using, 4-8, 4-9, 4-10, 4-17
- Jumping to DOS, 4-6
- Key code, 1-8
- LED indicators, 1-10
- List of devices, 1-9, 4-3, 4-4
- Listing the contents of memory, 5-11
- Loading a data file, 1-11, 4-13, 5-9
- Logic (PLD) functions, 4-1
- Logic menu, 3-2
- LPT ports, 1-7
- Main Optima screens, 1-9, 2-3, 3-1
- Manual conventions, 1-3
- Memory buffer, 4-3, 5-8, 5-11
- Memory contents, editing, 4-7
- Memory device, 1-9, 1-10, 1-14, 2-3, 3-6, 5-1
- Memory, listing the contents of, 5-11
- Messages, 1-3, 1-5, 1-7, 1-11, 1-21, 4-4, 7-1
- Multiple devices, programming, 4-14
- Multisyte unit, selecting, 4-14
- ON switch, 1-4
- Open dialog box, 1-11
- Operating the programmer, 3-2
- Optima / Optima-Light / Multisyte option, 1-4
- Optima PAL, 1-12
- Optima PROM, 1-9
- Optima PROM432, 1-14
- Optima release CD, 1-2, 1-4, 2-1
- Optima Test, 1-5
- Optima-Booster module, 1-7, 1-8
- Option bits, 1-9
- Options, command line, 4-1, 5-1
- Otestw, ending the task, 1-5
- Packages, 1-10
- PAL emulation, 4-6, 4-18
- Parallel cable, 1-2, 1-3, 1-8
- Parallel ports, 1-3
- PASS indicator, 1-10, 2-3, 5-14, 6-2
- Performance, verifying, 1-5, 1-20, 2-2, 6-1
- Physical specifications, 1-2
- PLCC adapter, 1-10
- PLD commands, 4-5
- PldAsm, 1-5
- Plus 48, 1-1, 1-4, 1-9, 2-2
- Port booster module, 1-7, 1-8
- Ports, 1-3, 1-7
- Power cord, connecting, 6-1
- Power requirements, 6-1
- Powering up the programmer, 1-4
- Production at high-volume, 1-8
- Programmable logic device, 1-12
- Programmer, running, 3-2
- Programmer, setting up, 1-3, 1-4, 2-1
- Programming a device, 4-15, 4-16
- Programming a logic device, 1-12
- Programming a memory device, 1-9
- Programming a serial number, 5-16
- Programming commands, PLD, 4-5
- Programming commands, PROM, 5-5
- Programming failure, 7-1
- Programming menu, 3-6
- Programming multiple devices, 4-14
- Programming parameters, 5-13
- PROM commands, 5-5
- PROM menu, 3-6
- PROM Option screen, 5-13
- Reading a device, 1-10, 4-16, 5-14
- Release CD, 1-2, 1-4, 2-1
- Requirements for power, 6-1
- Requirements for your computer, 1-2
- Running commands, 1-19, 3-1, 4-4, 4-6
- Running Self-test & Calibration, 1-5, 1-20, 2-2, 6-1
- Running the programmer, 3-2
- Scanning for JEDEC files, 1-19
- Scroll control, 1-19
- SCSI ports, 1-7
- Sector protection bits, setting, 1-9
- Security bit options, 1-9
- Security protection, setting, 4-16
- Selecting a device automatically, 5-3
- Selecting a device manually, 1-9, 4-3, 4-4
- Selecting a file, 1-18, 3-3, 4-13, 5-9, 5-10, 5-11
- Selecting a multisyte unit, 4-14
- Selecting the translation format, 1-11, 4-13, 5-9
- Self-test & Calibration, running, 1-5, 1-20, 2-2, 6-1
- Serial number, programming, 5-16, 5-17
- Set-programming, 1-14
- Set-programming options, 5-15
- Setting an address range, 5-5
- Setting up security protection, 4-16
- Setting up your computer and programmer, 1-3, 1-4, 2-1
- Singlesyte, 1-1
- Slow printer message, 1-7
- Software, installing, 1-4, 2-1, 2-2
- Supported devices, 1-2
- Supported file formats, 5-11
- System components, 1-2, 1-10
- System features, 1-2
- System, configuring, 1-3, 1-4, 2-1, 2-2
- Test (Otestw), ending, 1-5
- Test vectors, editing, 4-8, 4-9
- Testing a device using JEDEC, 4-8, 4-9, 4-10, 4-17
- Testing a device's integrity, 1-20
- Testing and calibrating the programming system, 1-5, 1-20, 2-2, 6-1
- TOP, 1-1, 1-9, 1-16
- TOP adapter, 1-10

- TOP432, 1-16, 2-3, 5-15
- TOP432DIP, 1-14
- Translation format, selecting, 1-11, 4-13, 5-9
- Translation formats that are supported, 5-11
- Turning on the programmer, 1-4
- Typical* installation option, 1-4
- UnAsm, 1-5
- Upgrading Optima Windows, 1-8
- Vendor, 1-9, 4-2, 5-2, 5-3
- Verifying a device, 4-17
- Verifying system performance, 1-5, 1-20, 2-2, 6-1
- Wall chart, 4-3
- Welcome to Data I/O Windows Setup dialog box, 1-4, 2-1
- Windows can start only one copy message, 1-5
- Writing buffer data to disk, 5-16
- Writing data to JEDEC file, 4-17

