USER'S MANUAL

ALL-07

PC-Based UNIVERSAL

Programmer and Tester

Manual V 2.0
June 1996

HI-LO SYSTEMS

Model numbers referred in this manual are:

Printer Port version
- Programmer : ALL-07, ALL-07A
- Interface card : PRN-07, PRN-07A

ISA Bus Card version
- Programmer : ALL-07PC
- Interface card : SAC-07, SAC-07CE

PACK option
- PAC-DIP40/48 : Universal DIP PACK
- PAC-EP32-8A/8B : EPROM 8 gang PACK

CONTENTS

4. FUNCTION REFERENCE GUIDE

1. INTRODUCTION

The manual provides instructions for installing and operating the Universal Programmer with an IBM PC or compatible running MS-DOS or PC-DOS. The Universal Programmer can operate with optional PACKs and adapters. PACKs with multiple sockets can perform gang programming of devices for mass production. Operation is similar regardless of which PACK is installed. For most examples in this manual we use the PAC-DIP40/48.

1.1 Programmer and Accessories

Your package should contain the following items:

* Universal Programmer.
* An interface Card.
* 1-M. cable with two 25-pin, D-type connectors on cable's end.
* Power cord (only for printer port version)
* DIAG-40 Diagnostic POD
* User's manual.
* Options: programmer PACK and software diskettes
  e.g. PAC-DIP40/48 with S/W diskettes
  PAC-EP32-8A/8B with S/W diskette ....

2. INSTALLATION

The UNIVERSAL programmer is easy to install and is good to work with an IBM PC386/486/Pentium computer or compatible running MS-DOS or PC-DOS. You are assumed familiar with the installation of PC add-on cards and software in PCs runnig MS-DOS.

2.1 Host System Requirements

* IBM PC386/486/Pentium or compatible PC.
* Max. system clock speed is up to 166 MHz.(A 166mhz Pentium is the fastest PC tested at the time of printing.)
* Min. 640K-byte memory.
* Min. 1 floppy disk drive and a hard disk with min. 6M Bytes free disk space.
* Operating system: MS-DOS or PC-DOS, version 3.3 or later.
* Release of TSR programs and EMM386.sys before running the S/W
* Do not run the S/W under Windows 3.x/95/NT DOS box.
* Do not let Green PC's enter power down or idle mode.
* Select EPP as the printer port mode if there are choices available.
* Select EPP as Unidirectional if there are choices available.

2.2 Software Installation Procedures

It is recommended to make a copy of the original software diskettes before installation. This is just a precaution in case the original diskettes are lost or damaged.

To install the Universal Programmer you first need to copy the supplied software to a subdirectory on your hard disk. The following example loads the software to a subdirectory called "PT":

Steps:	Description
C :\>md PT	To generate a programmer subdirectory (PT)
C :\>cd PT	To change directory to PT
C :\PT>copy A :*.*	To copy all files in A : to the current directory in C :

Note:

> PT is only an example name. Any name can be used for the subdirectory.

To install software updates in the future use exactly the same procedure. The new software will overwrite the existing programmer software drivers on your hard disk.

Software structure:
The software installed on the PC will contain the ACCESS.EXE shell program, *.dev menu device list files, individual device driver files, and some extra utility programs. ACCESS.EXE is the main system program. The ACCESS.EXE program provides a way to easily select a device manufacturer and device type and execute the device specific driver files used to program a device. All of the driver files can also be executed independently without running ACCESS.EXE. Driver files usually cover a group of related devices. For example the driver file EPP512.EXE supports programming of 2716-27C512 EPROM.

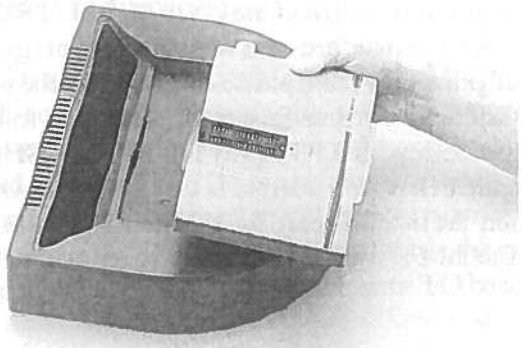
2.3 Hardware Installation Procedures

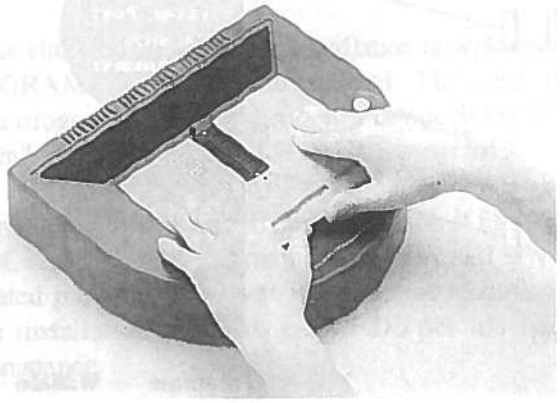
Before installation your programmer and PC should be turned off. The following guidelines should be followed during installation and during regular operation to insure a long productive life of the programmer:

Power on - first PC, then the programmer.
Power off - first programmer, then PC.
Never quit the device driver software while the programmer is in operation (Busy LED is on).

Step 1:

Before the Universal Programmer can operate a PACK option needs to be installed. To install a PACK first switch off your programmer. Next, insert the programmer PACK into the programmer slot near the antistatic sponge. The lip that extends from the bottom of the PACK should fit into the slot on the programmer. Press down the PACK to sit firmly as in the following picture.

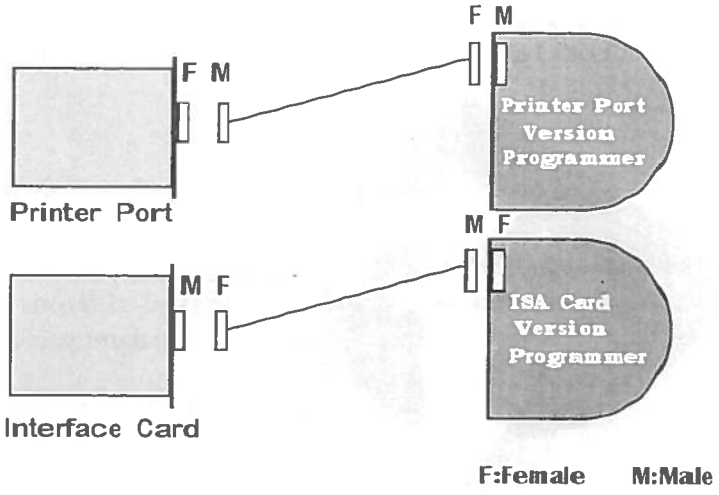




Step 2:

Connect the programmer to the PC using the included 1 meter D-25 cable. The programmer has a 25 pin male connector towards the back of the unit for printer port versions of the programmer and a 25 pin female connector for the ISA bus card version of the programmer.

* A printer port version programmer can be connected to the PC printer port directly. (Note, usually the I/O address for PC printer port is at 3BCH or 378H.)
* An additional printer port card is also included with the programmer. There is also an ISA bus interface card version of the UNIVERSAL PROGRAMMER. If you have the ISA bus interface card version of the programmer or if you use the additional printer port card please check and set the jumper for the interface card I/O address. The interface cards come with following three I/O address jumper selections: LPT1 (3BCH), LPT2 (378H), LPT3 (278H). The default H/W I/O address is LPT3(278H). Once the address is selected insert the interface card gently into an ISA bus slot in your PC, and fasten it to the PC frame with the slot cover screw.



Step 3:

First, power on the computer, then the programmer. The programmers on/off switch is located next to the 25 pin connector used to connect to the D-25 cable. For printer port version programmers, the power cord must be connected from programmer to the AC power source beforehand. The power supply of the programmer can accept 90 VAC - 270VAC, 47 Hz - 65 Hz power sources.

Secondly, check LEDs on the programmer.

ON LED must be ON.
BUSY LED must be OFF.
GOOD LED is in random state.

If the LEDs are not in the correct state, turn off the PC and the programmer and check all connections between the printer port or interface cards and the programmer. If the additional printer port card or the ISA bus interface card is being used then check that the card is securely installed in the ISA bus slot in the PC. Then turn on the computer and the programmer to check the LEDs on the programmer again. If the LEDs are still not in the proper state you may have another peripheral on the PC set to the same I/O base address.

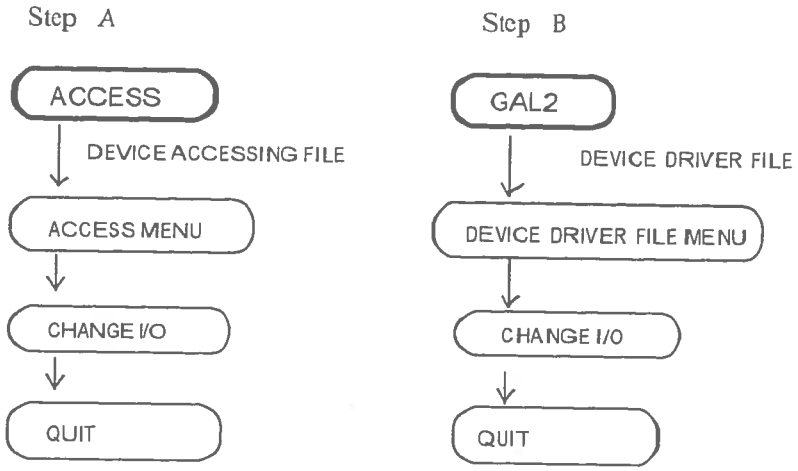
Step 4:

A final check of the hardware installation is performed when the UNIVERSAL PROGRAMMER software is executed. The software checks for the existence of the programmer H/W whenever a device driver file is run or from a special "Setup" function in the ACCESS.EXE program..

The hardware identification check tells you if H/W is successfully installed in the PC and if communication between H/W and S/W allows all functions to be operated properly. If the H/W cannot be identified by the S/W, that means some installation problems exist. Do not run the programmer under this circumstance.

The hardware identification test can be performed in two ways:

1. Run the ACCESS.EXE file
2. Run a Device Driver file

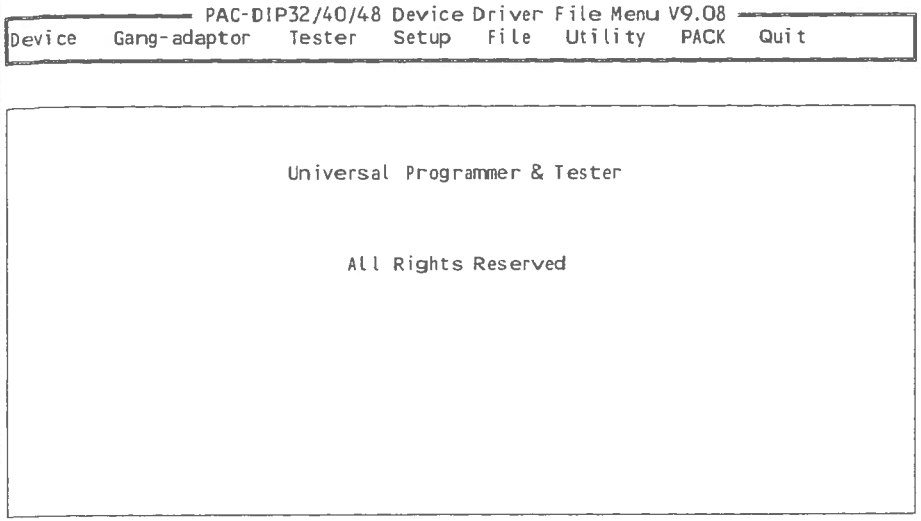


2 Ways to check hardware identification status

Step 4A:
Hardware identification from ACCESS.EXE menu.

Step 4A-1
Change to the subdirectory where you installed the programmer's software and type the 'ACCESS' command at the DOS prompt.

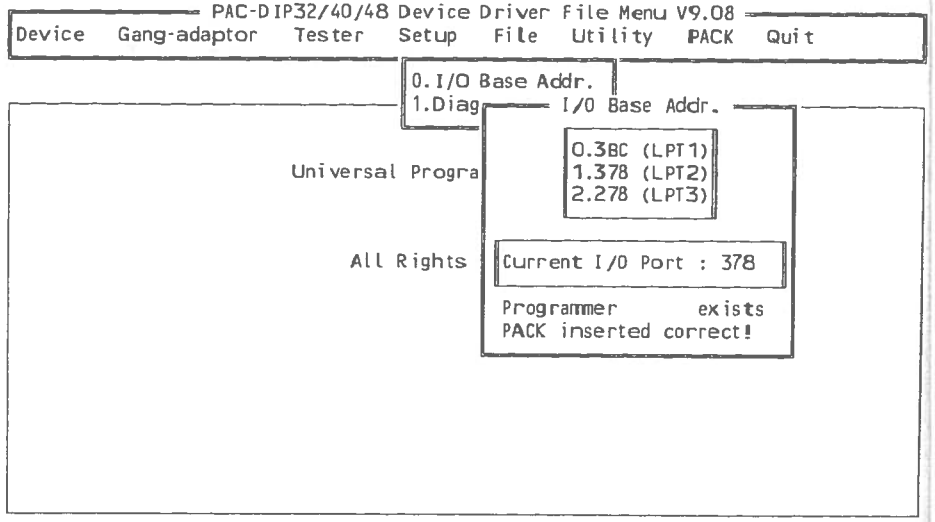
C:\PT\ACCESS <enter>



Access Main Menu

The very top line of the ACCESS menu will display the current PACK module selected in software. If this display does not match the PACK installed on the programmer then go to the "PACK" menu item on the right side of the screen and choose the PACK option that matches the hardware that is currently installed.

Once the PACK selection matches the hardware that is installed type letter "S" and "0" to enter the "I/O Base Addr." function.



i/o selection menu

The current I/O base address registered in the software will appear in the screen, and the software will automatically check with hardware identification circuits in the programmer. The software will display "Programmer Exists" or "Programmer Does Not Exist". When "Programmer Exists" appears, that means the hardware installation is OK.

Step 4A-2

The "Programmer Does Not Exist" message appears when the hardware I/O address of the interface card does not match the I/O address selected in software, when the hardware I/O address of the interface card conflicts with other add-on cards inserted in the PC, if the cables and interface card are not connected properly, or if the programmer is not turned on.

If the "Programmer Does Not Exist" message appears please try the following to resolve the problem:

1. Make sure the programmer is turned on and the D-25 cable is tightly connected to the programmer and the PC. Enter the listed I/O addresses: 0 (3BCH), 1 (378H), 2 (278H) sequentially until the"Programmer exists" message appears.

2. If the 1st solution fails, press "Q" to quit to DOS prompt, and then turn off the computer. Change the jumper address on the interface card to any of three choices for next identification check (repeat from Step 3 down to 4A-2, solution 1). If you still have problems please call your local sales representative for assistance.

After the Identification check passes OK and you quit to DOS the I/O address that was just set will be saved as the default address for the next time the software is run.

* Refer to Section 4.6 for more details about change I/O address function.

NOTE

Most problems are due to poor connections between the PC interface card and the PC slot, or improper cable connections between the PC interfacecard and the programmer. If you attempt to use a long parallel cable, 5-6 feet or longer, you may also experience the "Programmer does not exist!" problem.

Step 4B :

Hardware identification through Device Driver file (for example, GAL2.EXE).

Step 4B-1

Go to subdirectory PT and type the following command to run the GAL2.EXE driver file for GAL programmable logic devices.

```
C:\PT\GAL2 <enter>
```

NOTE

| GAL2 is one of the GAL device driver files. In the ACCESS function |
| the relevant driver file will be called after the type selection is entered. |
| The file can also be called separately for execution as in this example. |

The device driver file will be executed and will look for the parameter file GAL2.DAT. If it exists on the default disk drive, the parameter of I/O address, Manufacturer, Type and Programming Speed will be loaded as your default state. Otherwise, the default value I/O address (278), Manufacturer (AMD), Type (PALCE16V8H/Q/4/5), Programming Speed (standard) will be considered as the default state.

Once a device driver file is entered the hardware identification will be rechecked; if passed, that means the S/W I/O address matches to the H/W I/O address and the hardware installation is OK for programming. If the hardware is identified the following main menu will be displayed:

```
Universal   Programmer                  * Mfr.: AMD
                                        * TYPE: PALCE16V8H(Q)/4(5)/PALLV16V8
GAL2  section  V3.55
-----------  Main Menu  -------------    ┌──────────────────┐
1. DOS SHELL                            │    TARGET ZONE   │
2. Load JEDEC file to buffer            │ Check Sum : 0000 │
3. Save buffer to disk                  └──────────────────┘
4. Edit buffer     7. Display buffer
5. Change I/O base address

T. Type select     M. Mfr. select

B. Blank check     D. Display
P. Program         A. Auto(B&P&V&S)
R. Read            V. Verify
E. Erase           S. Security fuse blow
Q. Quit
-------------------------------------


Select function ?
```

Main Menu

Step 4B-2

If the following error message appears on the screen:

```
Error identification on hardware!
Press "Q" to quit,
or press < CR > to continue
```

The hardware I/O address of the interface card does not match the default software I/O base address, there are conflicts with other add-on cards or peripherals inserted in the PC, or the programmer power is not turned on.

There are two solutions in turn to solve the problem.

1. Press < enter > to enter the main menu of Device driver file, then enter function 5. 'Change software I/O address' followed by selecting the listed I/O addresses: 0 (3BCH), 1 (378H), 2 (278H) sequentially till "Programmer exists" message appears.

2. When the 1st solution fails, press "Q" to quit to DOS prompt, and then turn off the computer. Change the jumper address on the interface card to any of three choices for next identification check (repeat from Step 3 down to Step 4B-2, solution 1). If you fail with both ways, the programmer needs maintenance.

After the Identification check is passed and quitting to DOS the I/O address set will be saved as default of next entry.

* Refer to Section 4.6 for more details about change I/O address.

NOTE

| 1. Most problems are due to poor connections between the |
| interface card and the PC slot, or poor cable connections |
| between the interface card and the programmer. |

2. For a printer port version programmer, the programmer can be connected to the PC printer port which usually has its I/O address set at 378H. Since the factory default I/O address in S/W is 278H, the S/W address needs to be changed through the "Change I/O base address" function to match the printer port address of your PC. (see Section 4.6 for more details).

3. USING THE PROGRAMMER

Before starting to use the programmer definitions of some commonly used symbols are explained here for your reference.

3.1 Definition of Key Symbols and Terms

3.1.1 Device Symbols for EPROM, EEPROM, BPROM and MPU

* **Bit, Nibble, Byte, Word, Address and Buffer Base Address**

  Bit : A basic unit of Binary data.

  Nibble : A group of 4-bit Binary data. A nibble's value ranges from 0H to FH.

  Byte : A group of 8-bit Binary data. A byte's value ranges from 0H to FFH.

  Word : A group of 16-bit Binary data. A word's value ranges from 0H to FFFFH.

  Address : A hexadecimal number used to represent the location of data. Address values generally ranges from 0H to FFFFFH.

  Buffer Base Address : The actual physical address of the buffer in PC memory. Buffer Base Address values range from 0000 : 0000 to F000 : FFFF.

* **Working Buffer**

  In the default state, the working buffer is a block area of PC memory allocated by the device driver file through DOS. The working buffer is used by the device driver file as an intermediate storage.

  When you want to program a data file to a DEVICE you first load the file to the working buffer. The contents in the working buffer are then programmed to the target DEVICE. When you read the contents of a master DEVICE the data is stored in the working buffer. The working buffer can then be edited or saved to disk for future reference.

The minimum allocated size of the working buffer is 64K bytes, and the maximum size is the maximum available memory in the PC (This does not include expanded or extended memory. Please see the following section to overcome this limitation.)

Since the working buffer is dynamically allocated through DOS, the actual base address on the PC may vary from system to system, software to software. The user need not refer to the actual base address of the working buffer for most practical purposes.

NOTE

> In the default state the PC memory buffer is used as the working buffer. PC memory is limited to just 640K. This limit can be overcome by using the "Modify buffer structure function to specify hard disk C:, D:, E:, or F: as the working buffer. You can also specify a RAM drive as the buffer area. (See 4.19 Modify buffer structure)

* **Buffer Start and Buffer End Addresses**
The buffer start and end addresses are offset addresses specified from the base address of the WORKING BUFFER. The data stored in the WORKING BUFFER between the buffer start and buffer end addresses contains the subset of the working buffer that is programmed to a target device. This is also the area of the buffer used for checksum calculations.

* **Check Sum**
This is the sum of all data contents between buffer start and end addresses. All bytes are added byte by byte and then the least significant 16 bits (4 HEX characters) are displayed as the check sum. Some bytes in the address range may not be included in the checksum as specified by the manufacturer of the particular DEVICE. Check sums will be calculated during the DEVICE reading, file loading, type changing or after buffer editing.

Bit count of the data contents:
NIBBLE WIDE PROM is 4.
BYTE WIDE PROM is 8.
WORD WIDE PROM is 16.
MPU is 8.

Device Start and Device End Addresses
The start and end addresses are offset addresses specified by the DEVICE. Data stored at the buffer start address in the working buffer will be programmed to a device at the Device Start Address location.

* **EVEN and ODD address mapping**
The address sequence of data contents can be assigned to CONTIGUOUS, EVEN or ODD whenever you want to READ, PROGRAM or VERIFY the EPROM.

For example, in a program function submenu the software will ask you:

Ready to start (Y/Even/Odd/<CR>)?

You may press Y to program the data in the buffer to the device CONTIGUOUSLY as follows :

Buffer start	+ 0	to Device start	+ 0
	+ 1	to	+ 1
	+ 2	to	+ 2
	...	to	...

You may press E to program the data in the EVEN address to the device as follows:

Buffer start	+ 0	to Device start	+ 0
	+ 2	to	+ 1
	+ 4	to	+ 2
	...	to	...

You may press O to program the data in the ODD address to the device as follows:

Buffer start	+ 1	to Device start	+ 0
	+ 3	to	+ 1
	+ 5	to	+ 2
	...	to	...

In the READ operation, the software will perform in the reversed direction. Even/Odd address mapping is used for 16 bit systems that use 2 memory devices to store program code.

* **I/O Address**
This is the I/O base address of the interface card. Each I/O interface card added into a PC slot will occupy one or more I/O addresses.

The default I/O base address of the interface card is 278 and occupies 3 contiguous spaces (278 to 27A).

* **Counter**
This is the programming address counter. During the DEVICE programming, the counter on the screen will scroll to show the current address in the target device that is being programmed.

* **MFR, TYPE, VPP, SPEED**
Every DEVICE has its own manufacturer (MFR), type number (TYPE), programming voltage (VPP) and programming speed (SPEED or algorithm). Please refer to Chapter 4 for detailed descriptions.

3.1.2 Device Symbols for special devices PLD, PAL, FPL, GAL, PEEL, CPLD, EPLD and FPGA.

* **Programmable Logic Device (PLD)**
Generally speaking, a device that can be programmed to perform many different logic operations is a PLD.
PLDs are grouped into following five categories:
PLD : A one time PLD such as a PAL device.
EPLD : An UV erasable PLD such as a EPLD, CPLD, or FPGA device.
EEPLD : An electrically erasable PLD such as a GAL, PEEL, CPLD, or FPGA device.
CPLD : A much more complex PLD device.
FPGA : Field programmable gate array.

* **JEDEC Fuse Map File of a PLD Device**
The JEDEC fuse map file is the standard format which can be loaded by the programmer. It contains fuse information (Blown/Intact) and the Function Test Vector of the PLD. Most PLD assemblers or compilers, such as the PALASM, PLAN, OPAL, CUPL, ABEL, AMAZE and PDK-1, will produce a JEDEC fuse map file.

* **POF fuse map file of PLD device**
The POF fuse map file is a format used for programmable logic devices from Altera Corporation. POF files store fuse data in a more compact way.

* **Fuse Blown and Intact**
A new programmable device generally comes with all fuses in an intact state(blank). The designer can only blow an Intact fuse into a Blown state i.e. 0 state. The default state of a new device may be opposite to this depending upon the technology used to manufacture the device. For one time programmable (OTP) devices you can not change the state of a fuse back to the unprogrammed (default) state once it is blown (programmed). UV erasable devices have a window on them. When this window is exposed to UV light for some time the fuses will be erased to the default (or unprogrammed) blank state. Other devices are electrically erasable and can be erased by

using a function on the Universal Programmer.

* **Array Fuses, Configuration Fuses**
  Array fuses are the main logic fuses in a PLD. Different arrangements (Blown/Intact) will have different logic combinations.

  Configuration fuses always indicate the I/O architecture of the PLD, such a Combinatorial/Registered, Output Feedback/Output Enable.

* **Security Fuses**
  Security fuses are present in many PLD/CPLD devices and microcontrolle devices. When a security fuse is programmed the data stored inside the PLI or microcontroller can not be read by a device programmer for revers engineering purposes. Generally a device will read as blank if the securit fuse is blown. The device will operate functionally equivalent with the securit fuse intact or blown.

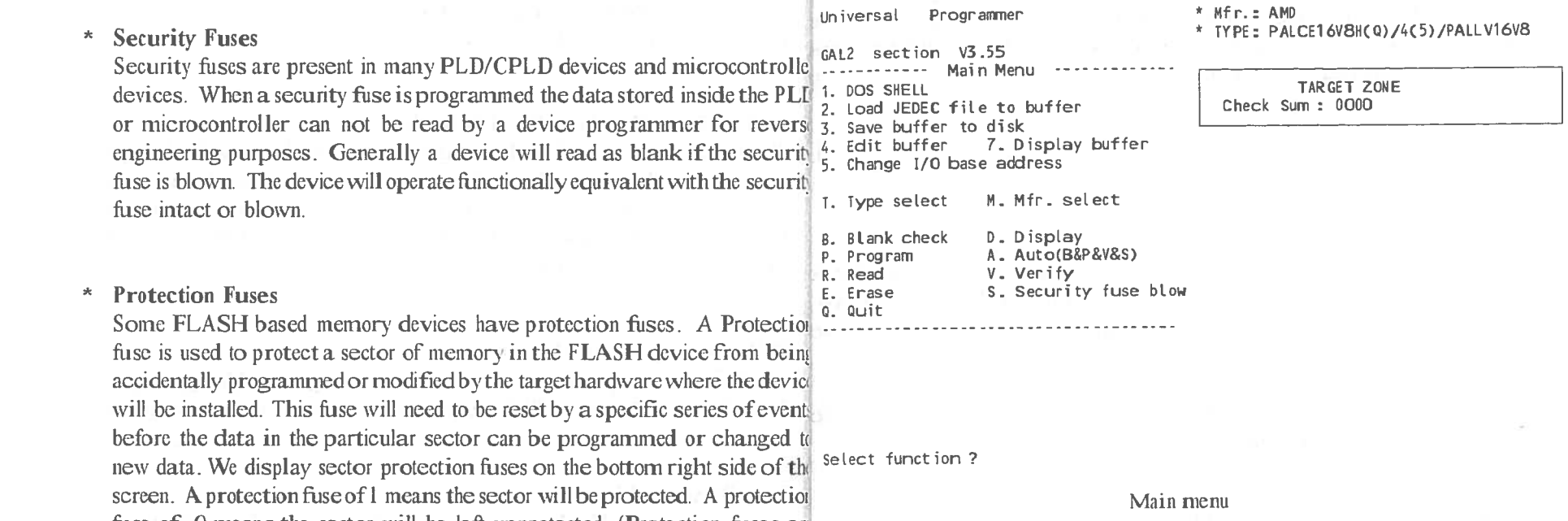
* **Protection Fuses**
  Some FLASH based memory devices have protection fuses. A Protectio fuse is used to protect a sector of memory in the FLASH device from bein accidentally programmed or modified by the target hardware where the devic will be installed. This fuse will need to be reset by a specific series of event before the data in the particular sector can be programmed or changed t new data. We display sector protection fuses on the bottom right side of th screen. A protection fuse of 1 means the sector will be protected. A protectio fuse of 0 means the sector will be left unprotected. (Protection fuses ar only programmed using the separate "Protection" function or the "Auto function. Please see the function descriptions section of the manual for mor details.)

3.2 Viewing the Main Menu of a Device Driver file

Before we begin operation please take a moment to familiarize yourself with the main functions found in the device driver files. Device driver files contain all of the device specific functions.

3.2.1 Main Function Menu

All main functions are listed on the left side of the screen.



Main menu

3.2.2 Status Field

In the upper right window of the screen is a group of text about the target device being used including MFR. TYPE, VPP, Programming speed, working buffer and device address. This is called status field.

Whenever the programmer is being used, please make sure that data in the

status field matches your requirements. Otherwise the target DEVICE may b damaged or programmed to an unknown state.

3.2.3 Logo, Hardware Model and Software Version

On the upper left of the screen, you will see the LOGO, HARDWARE MODEL and SOFTWARE VERSION. These messages are for reference only.

By now you should be familiar with the main menu screen. You may learn mor about these features by reading the rest of this section. All relevant function will be discussed in detail in later chapters.

3.3 Getting Started

The following example will demonstrate how to start and exit the device drive main menu through ACCESS and use functions such a MFR, TYPE, LOAD BLANK CHECK and PROGRAM. If you do not get the desired results, pleas double check your installation.

In the example, we are going to use an AMD PALCE16V8H GAL. Th procedure for working with other types of devices is similar.

3.3.1 Starting from the ACCESS file.

3.3.1-1 Execute the ACCESS program.

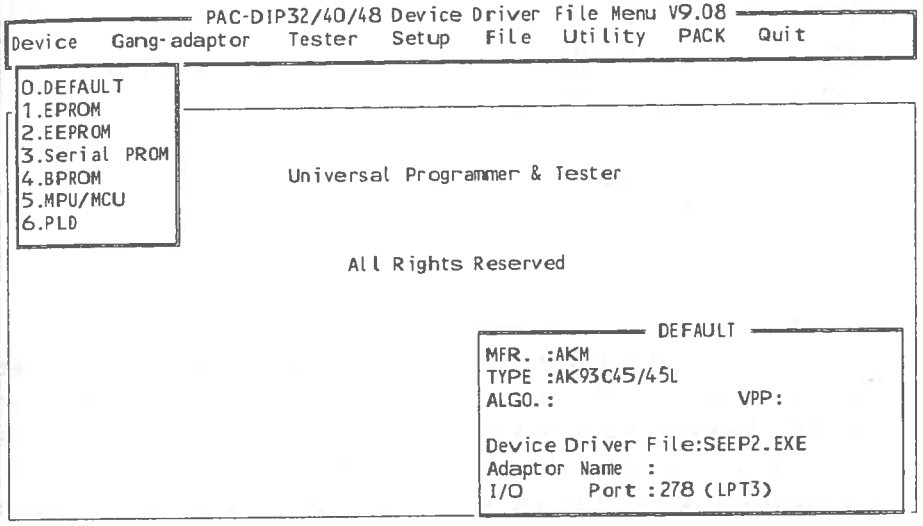
Change to the subdirectory where you installed the programmer's software and type the "ACCESS" command at the DOS prompt.

```
C:\PT\ACCESS
```


After entering the "ACCESS" command, ACCESS.EXE will load the PROG.DEV device list information file and the following ACCESS MENU will appear.

On top of the ACCESS menu the current PACK selection , "PAC-xxxxx", will be displayed. If this display does not match the actual PACK that is installed on the Universal Programmer then use the PACK menu item on the far right of the screen to select the correct PACK option.p This section of the manual is using the PAC-DIP40/48 as an example.

Select letter "D" to go to the "DEVICE" menu item, and select "6" for the PLD group. The following MFR list will appear on the screen:



Press <ESC> to previous menu or '*' to main menu.

3.3.1-2 Select PLD MFR and TYPE.

Select the desired manufacturer, in this case AMD, by moving the highlighted bar to the AMD position and pressing the < ENTER > key. Alternatively you

can type the number displayed in front of the desired manufacturer and press the < ENTER > key. Then the type menu screen will appear.

```
PAC-DIP32/40/48 Device Driver File Menu V9.08
Device   Gang-adaptor   Tester   Setup   File   Utility   PACK   Quit

 0.DEFAULT
   MFR.                    PLD Device          TYPE
00.ALTERA          000.AmPAL16R4
01.AMD             001.AmPAL16R4A
02.AMI             002.AmPAL16R4AL
03.ATMEL           003.AmPAL16R4B/Q/L
04.CYPRESS         004.PAL16R4D/2
05.GOULD           005.PAL16R4-4              *CNV-PLCC-16L8-4
06.HYUNDAI         006.PAL16R4-5/-7
07.ICT             007.PAL16R4H-10
08.INTEL           008.AmPAL16R6
09.LATTICE         009.AmPAL16R6A
10.MMI             010.AmPAL16R6AL
11.NS              011.AmPAL16R6B/Q/L
12.PHILIPS         012.PAL16R6D/2
13.RICOH           013.PAL16R6-4              *CNV-PLCC-16L8-4
14.SAMSUNG         014.PAL16R6-5/-7
15.SEEQ            015.PAL16R6H-10

Select Mfr. number:
Press <TAB> to switch window or <ENTER> to select.
Press <ESC> to previous menu or '*' to main menu.
```

Type selection

The device type is selected in the same way as the manufacturer. You can use the arrow keys to highlight a particular device or you can type in the number in front of the desired selection followed by pressing the < ENTER > key. In this case you can type "32" followed by pressing < ENTER > to select PALCE16V8H. After selecting the correct type, the ACCESS program will automatically search for the relevant device driver file on the current subdirectory and then ACCESS will execute this file. In this case the GAL2.EXE driver file will be executed.

The main function menu of that device driver file will then be displayed. To continue this example please go to Section 3.3.3. Section 3.3.2 shows an alternative way to run a device driver file and select a device manufacturer and type without using the ACCESS menu system. This can be useful for batch file operation which is explained later.

3.3.2 Starting the Device Driver file without using ACCESS.

3.3.2-1 Execute the proper device driver file.

The proper device driver file required for a particular device can be obtained by running the ACCESS.EXE program and using the "Cross Ref." function under the "FILE" menu item.

For the PALCE16V8H, please execute the device driver file "GAL2.EXE" by typing the following text:

```
C:\PT\GAL2 <enter>
```

The following main function menu will now appear on the screen:

```
Universal   Programmer                    * Mfr.: AMD
                                          * TYPE: PALCE16V8H(Q)/4(5)/PALLV16V8
GAL2  section  V3.55
------------  Main Menu  -------------     ┌──────────────────────┐
1. DOS SHELL                              │      TARGET ZONE     │
2. Load JEDEC file to buffer              │ Check Sum : 0000     │
3. Save buffer to disk                    └──────────────────────┘
4. Edit buffer     7. Display buffer
5. Change I/O base address

T. Type select     M. Mfr. select

B. Blank check     D. Display
P. Program         A. Auto(B&P&V&S)
R. Read            V. Verify
E. Erase           S. Security fuse blow
Q. Quit
-------------------------------------



Select function ?
```

3.3.2-2 Select PLD MANUFACTURER and TYPE.

Type "M" to display a list of device manufacturers supported by the devic driver file. Make a selection by typing the number located in front of the desire manufacturer and pressing the < ENTER > key.

```
Universal   Programmer                    * Mfr.: AMD
                                          * TYPE: PALCE16V8H(Q)/4(5)/PALLV16V8

GAL2  section  V3.55
-----------  Main Menu  ------------
1. DOS SHELL
2. Load JEDEC file to buffer
3. Save buffer to disk
4. Edit buffer     7. Display buffer
5. Change I/O base address

T. Type select     M. Mfr. select

B. Blank check     D. Display
P. Program         A. Auto(B&P&V&S)
R. Read            V. Verify
E. Erase           S. Security fuse blow
Q. Quit
-------------------------------------

Select function ? m
```

```
        TARGET ZONE
  Check Sum : 0000

 Mfr. SELECT :
0. AMD

<CR> back to main menu.
SELECT NUMBER ?
```

Submenu of manufacturer

When the manufacturer is entered, the name in the status field (upper right corner of the screen) will be updated. Then press < ESC > or < enter > to clear the manufacturer selection box and return to the main menu. Now, type "T" from the main menu. The screen should display a list of available types for the selected device manufacturer as follows:


```
Universal   Programmer                    * Mfr.: AMD
                                          * TYPE: PALCE16V8H(Q)/4(5)/PALLV16V8

GAL2  section  V3.55
-----------  Main Menu  ------------
1. DOS SHELL
2. Load JEDEC file to buffer
3. Save buffer to disk

Select function ? t
```

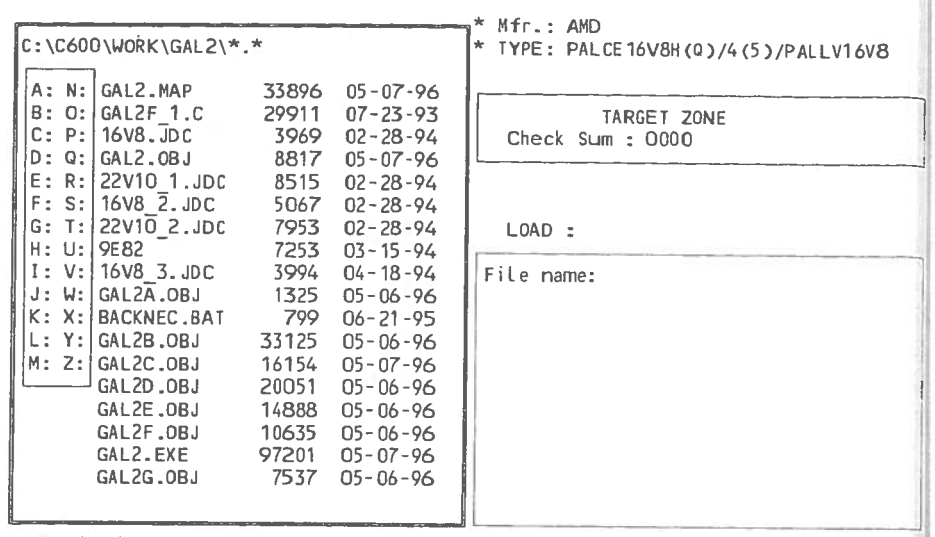
```
        TARGET ZONE
  Check Sum : 0000

 TYPE SELECT :
00.PALCE16V8H(Q)/4(5)/PALLV16V8   08.PALCE22V10H(Q)/4
01.PALCE16V8Z/PALLV16V8Z          09.PALCE22V10H(Q)/5/PALLV22V10

02.PALCE20V8H(Q)                  10.PALCE16V8HD

03.PALCE22V10H(Q)                 11.PALCE20RA10H(Q)/4
04.PALCE22V10Z/PALLV22V10Z
                                  12.PALCE20V8H(Q)/4(5)
05.PALCE610H

06.PALCE29M16H/4
07.PALCE29MA16H/4

  <CR> back to main menu.
Select which number ?
```

Submenu of Type

Type in the number corresponding to the desired type , (e.g., 0 for 16V8H). The type will automatically be updated and displayed in the status field.

3.3.3 Load Disk File into Buffer

After selecting a device manufacturer and type you are ready to begin working with a device. Usually you will want to program a device with a data file you have stored on disk or you will want to read the contents of a master device and copy the contents from the master to another device. We will cover both alternatives in this example. In this example we are working with a PLD/GAL device which requires a JEDEC fuse map file as input. Press "2" to select the "Load JEDEC file to buffer" function and the following file loading submenu will appear:



Submenu of file loading

Type in the name of the file that you want to transfer to the working buffer. The drive letter and any path names must be included.

An alternative to the direct typing method is to select the intended file from the directory window shown on the left side of the screen. Please follow the listed steps to select the file from the window selection.

- When prompted for a file name press the < TAB > key to change the active window from the load window to the directory window.

- Use the <UP>, <DOWN> arrow keys or the <PgUp>, <PgDn>, <Home>, <End> keys to move the highlighted bar to the desired file or subdirectory.

- When a subdirectory name is highlighted, the directory window will show all the files included in that subdirectory for selection. Pressing the

< F1 > function key will let you enter a file mask such as "*.JED" so only the files you are interested in are displayed for selection.

Once the correct file is highlighted press the < ENTER > key to select the desired file.

After selecting a file the active window will automatically change back to the "Load:" window, and the selected file name will appear on the "File Name" line.

After the file name has been entered, the software will begin to load the disk file to the working buffer and display:

```
Loading now...
Ok!
```

This message means the disk file is now in the working buffer. A file in the working buffer can be programmed to another device or edited and saved back to disk. Press < ESC > or < ENTER > to return to the main menu.

3.3.4 Read Contents from Master PLD

If the PLD data is in a MASTER PLD instead of a disk file, you can transfer the device contents to the working buffer by typing "R" to enter the READ function. After pressing R, the following submenu will appear:

```
Universal   Programmer                    * Mfr.: AMD
                                          * TYPE: PALCE16V8H(Q)/4(5)/PALLV16V8
GAL2  section  V3.55
----------- Main Menu ------------        TARGET ZONE
1. DOS SHELL                              Check Sum : 0000
2. Load JEDEC file to buffer
3. Save buffer to disk
4. Edit buffer    7. Display buffer
5. Change I/O base address

T. Type select    M. Mfr. select          READ to buffer :
                                          Ready to read (Y/<CR>)?
B. Blank check    D. Display
P. Program        A. Auto(B&P&V&S)
R. Read           V. Verify
E. Erase          S. Security fuse blow
Q. Quit
-----------------------------------

Select function ? r
```

Insert the MASTER PLD into the socket while the pull lever is up. Pin 1 should be located on the lever side of the DIP socket. If the device has less pins than the DIP socket then the device should be bottom justified in the socket. Close the lever. Then press Y, and the data of the MASTER PLD will be transferred to the working buffer and display:

Reading now...

Ok!

Press <ESC> or <enter> to return to the main menu.

3.3.5 Insert the Blank PLD into Socket

After transferring the data from the disk file or the MASTER PLD to the working buffer you are ready to program a new device. Take out the MASTER PLD and insert a blank PLD into the socket while the pull lever is up. Again, Pin 1 should be located on the lever side of the DIP socket. If the device has less pins than the DIP socket then the device should be bottom justified in the socket. Close the lever.

CAUTION

> The notch-end of the PLD must face the lever on the programmer's DIP socket. If the device has less than 40 pins hen the device should be bottom justified so the lower pins on the device (farthest from pin 1) are closest to the GOOD LED on the programmer. If the IC is inserted incorrectly the PLD may be damaged or programmed to an unknown state.

3.3.6 Program Buffer Contents to PLD

After loading the disk file or reading the MASTER PLD data into the working buffer, and inserting a blank PLD, the program function is used to copy the contents of the working buffer to a device. To invoke the programming function type. After typing P, the PROGRAM function will display the following submenu:

```
Universal   Programmer                    * Mfr.: AMD
                                          * TYPE: PALCE16V8H(Q)/4(5)/PALLV16V8
GAL2  section  V3.55
----------- Main Menu ------------        TARGET ZONE
1. DOS SHELL                              Check Sum : 0000
2. Load JEDEC file to buffer
3. Save buffer to disk
4. Edit buffer    7. Display buffer
5. Change I/O base address

T. Type select    M. Mfr. select          PROGRAM :
                                          Ready to program (Y/<CR>)?
B. Blank check    D. Display
P. Program        A. Auto(B&P&V&S)
R. Read           V. Verify
E. Erase          S. Security fuse blow
Q. Quit
-----------------------------------

Select function ? p
```

Submenu of Program

Press "Y" on the PC keyboard or the "YES" key on the Universal Programmer to start programming the buffer contents into the blank PLD. (If you are not sure if the device in the socket is BLANK please use the "B" function to test if the device is really blank before programming.) At the end of the programming process, the programmer will compare the PLD device contents with the working buffer. Any discrepancies will then be displayed.

The comparison finalizes the programming process. For programming other PLDs, wait until the BUSY LED turns off, then replace the PLD and type "Y" again.

If you want to exit the programming process, press < ESC > or < enter > to return to the main menu.

The operation to program a PLD is very simple and similar to the method used to program nearly all other types of devices. Many other functions and operations are available with this Universal Programmer. The details of all functions are described in the Chapter 4 FUNCTION REFERENCE GUIDE.

3.4 Expansion Adapters and Converters

The rapid development of Integrated Circuits has lead to a huge variety of device types and packaging techniques. We have developed a wide array of adapters and converters that allow the Universal programmer user to support virtually any IC type or device packaging that comes on the market. These adapters are designed with standard DIP footprints so a programmer with a 40 pin or larger DIP socket can program devices in virtually any package and with virtually any pin count. Devices with 8 pins to over 200 pins can be supported on the same programmer.

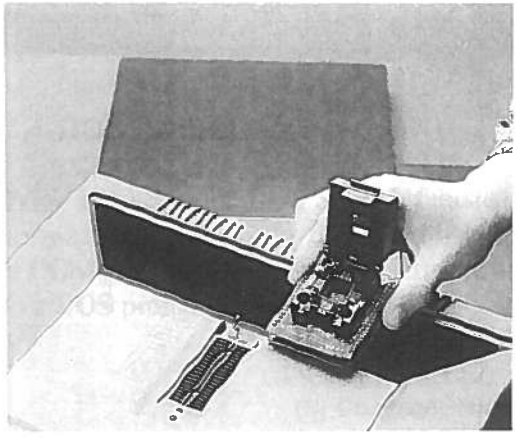
3.4.1 Adapter and Converter Installation

Adapter -
Each adapter has 40 pin DIP footprint and will work with both the PAC-DIP40 and PAC-DIP-48. Adapters support devices that have more than 44 pins or require some special programming requirements. Each adapter includes a software driver that should be copied into the same subdirectory as the ACCESS.EXE program.

Converter -
Each converter translates from a DIP footprint of a device to a surface mount footprint (PLCC, SOP, TSOP, etc...). Converters support devices with 8 pins to 44 pins and do not require any additional software.

S/W Installation -
As mentioned earlier, adapter software should be copied into the same directory as the PAC-DIP40/48 software drivers and ACCESS.EXE.

H/W Installation -
Insert the adapter or converter into the DIP socket of PAC-40/48 in a pin to pin way as shown in the following figure. Installing an adapter or converter is similar to placing a DIP IC into the DIP socket.



3.5 Special Single PACK and Gang PACK

For many devices the Universal Programmer provides more than one option for programming a device. For 44 and 68 pin PLCC devices you can use a device specific adapter or converter or you can use the general purpose PAC-PLCC44 and PAC-PLCC68. The PAC-PLCC44 supports programming most devices in a 44 pin PLCC package. The PAC-PLCC68 supports programming most devices in a 68 pin PLCC package.

For production requirements special multiple socket gang options are available for EPROM/EEPROM/FLASH, microcontrollers and microcontroller peripherals, Serial PROM, and programmable logic devices.

Dedicated S/W for each PACK should be copied to the subdirectory where ACCESS.EXE is located. Before application, please use the PACK menu item in ACCESS to select which PACK will be used. Next you select a device manufacturer and device type in same way as is done with the PAC-DIP40/48. Please note that on gang PACKs socket number 1 doubles as a master socket for reading master devices.

4. FUNCTION REFERENCE GUIDE

The following reference guide describes each of the functions found in the main menu of the Device Driver files. The functions will be described in the order as given in the menu. To enter into the desired function, press the letter or number preceding that function.

EPROM, EEPROM, BPROM and MPU Driver File menus

```
Universal   Programmer

EPROM 512  section  V3.53
----------  Main Menu  ------------
1. DOS SHELL
2. Load BIN or HEX file to buffer
3. Save buffer to disk
4. Edit buffer      7. Display buffer
5. Change I/O base address
6. Display loaded file history
9. Modify buffer structure

T. Type select      M. Mfr. select
Z. Target zone
S. Program speed,algo

B. Blank check      D. Display
P. Program          A. Auto(B&P&V)
R. Read             V. Verify
C. Compare & display error
Q. Quit
-------------------------------------
Buffer size      : 256K bytes
Buffer structure : PC MEMORY
Select function ?
```

```
*Mfr.:27/27C          *BLANK BIT: 1
*TYPE:x16             *PGMSPEED:INTL
*VPP :25V             *VCP :6.0V

              TARGET ZONE
Buffer start  addr.: 00000
      end     addr.: 007FF
      Check Sum  :   0000      COUNTER
Device start  addr.:   0000       0000
```

4.1 DOS SHELL

Press 1 to select the DOS SHELL function. After selecting this function, the software will look for the file COMMAND.COM at the DOS boot disk drive. If COMMAND.COM exists, the software will execute this file and pass control to a DOS prompt as follows:

```
Type EXIT to return to Programmer
Microsoft (R) MS-DOS (R)  Version 3.30
   (C) Copyright Microsoft Corp 1981-1987

C:\PT\EPP512>>
```

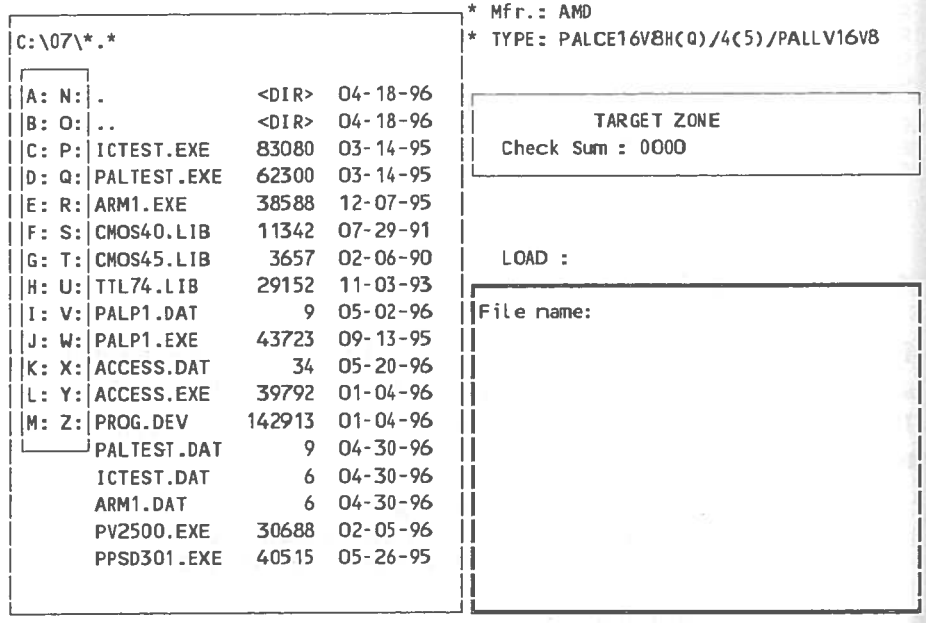
The S/W is now controlled by DOS and waiting for your commands. Under the DOS prompt you can run other programs or utilities and easily return to the programmer software at the same point you exited to the DOS shell previously. The command format under the DOS shell is the same as for the standard DOS prompt. Type "exit" and press < enter > to return to the main menu of the programmer software driver file..

NOTE

> This function will work only if the COMMAND.COM is already on the DOS boot disk drive.

4.2 Load BIN or HEX File to Buffer

Press 2 to select the load file function. A dialog window will be displayed on the screen as follows:



Files in the following three formats can be loaded to the working buffer when working with memory or microcontroller devices:

1. BIN file
2. INTEL HEX
3. MOTOROLA S HEX

When prompted for a file name, you should:

A) Enter the complete file name that you want to load including the path name if the file is not in the same directory as the programmer's driver files. or

B) Press the < Tab > key to switch control to the window on the left. When the window on the left is highlighted use the following commands to select a file for loading:

* < Tab > : the toggle switch to activate the left or the right window.

* < Page Up >, < Page Down >,< Up >, < Down >, < Home >, < End > : move the highlighted bar to a particular file or to a subdirectory where other files are located.

* < CR or Enter > : selects the file located under the highlighted bar to be loaded to the working buffer.

* < ESC > : exits the file loading menu.

* < Drive-letter > : type in a drive letter, for example, will display all of the files and directories located on that drive.

NOTE:

> These commands will only work when the Tab key is pressed to activate the left window.

Once a file has been selected to be downloaded to the buffer you will be prompted " < B >in, < I >ntel HEX, < M >otorola S HEX: ". You should select 'B' if the file you are loading is a binary file, 'I' if the file you are loading is a Intel HEX file, or 'M' if the file is in one of the Motorola S Record formats. If your assembler or compiler has the option to output binary files it is the best to use that option. Files already in binary format will be loaded faster. All HEX files are actually converted to Binary by the driver file before being loaded to the buffer.

If 'B' is chosen you will be prompted "Load Address(00000):", at this point you should press < CR or enter > on the keyboard if you want the file loaded at address 0 of the buffer memory, or type in the HEX number of the address where you would like the file to be loaded. The file will then be loaded to the buffer at the address you entered.

If 'I' is chosen you will be prompted for 'File start seg.(0000):, here the file start segment address is required. All data between the start segment X 16 and the end of the file will be converted to binary and loaded to the buffer memory. The segment range is from 0 to F000.

If 'M' is chosen you will be prompted for 'File start addr.(00000000):', here the actual start address of the file is required. All data between the given start address until the end of the file will be converted to binary and loaded to the buffer. The address range is from 0 to FFFFFFFFh.

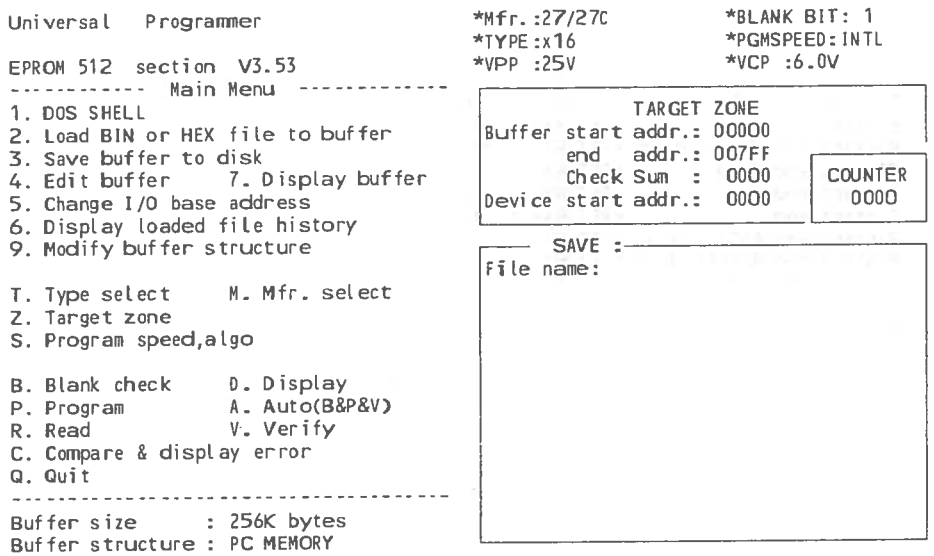
If 'I' or 'M' is chosen you will be prompted, 'Unused bytes will be < 1 >00 < 2 > FF < 3 > don't care:-'. In a typical application when a HEX file is loaded to an EPROM only a portion of the EPROM's available memory is actually filled with your program or data after being converted to binary. The rest of the unused portions of the EPROM may be filled with all 00's or all FF's depending on your preference. To do this choose '1' or '2' at the prompt. If you would like to load more than one HEX file to one ROM then you should choose '3' at the prompt. This means that the portions of the EPROM not used by the first HEX file loaded to the buffer can be filled with whatever other data you want by loading other HEX files and choosing the " don't care " option.

NOTE

Intel HEX files and Motorola S records contain address information in their formats. This means data may only be specified starting at an address greater than 0000h in some files. If a start segment address or file start address of 0000h is selected then all bytes between the start segment or file start address and the first address where data is specified in the HEX file will be filled with 00 or FF depending upon the response to the "Fill unused bytes...." prompt. Sometimes this offset can be larger than the address space of the buffer for the target device you would like to program and no data is loaded to the buffer except for "00" or "FF" bytes. In this case the correct start segment or file start address should be entered. This information should be specified by the firmware engineer that compiled the source file.

4.3 Save Buffer to Disk

Press 3 to select the save file function. A dialog window will appear on the screen as follows:



When prompted, type in the complete name under which you want the file to be saved including the drive letter and any path names and then press < enter >. You can save the entire buffer contents to disk or just a portion of the buffer. Prompts are displayed for the "Start Address" and "End Address". Please key in a hexadecimal address to respond to these prompts and then all the data in the specified address range will be saved to disk.

When the OK! message appears the function is completed. Press < ESC > to return to the main menu.

NOTE

Files saved under this function are Binary files with start and end addresses entered in byte addresses.

4.4 Edit Buffer

Press "4" to select the buffer editing function and to obtain the editing command summary on the screen as follows:

```
            EDITING COMMAND SUMMARY
===============================================
D [start],[end]          <RETURN> : DUMP
E start                  <RETURN> : EDIT
M start,end,destination  <RETURN> : MOVE BLOCK
F start,end,data         <RETURN> : FILL BLOCK
P start,end              <RETURN> : PRINT BLOCK
C start,end              <RETURN> : CHECK SUM
S start,end,ASCII data   <RETURN> : ASCII  SEARCH MAX. 15 characters
B start,end,BINARY data  <RETURN> : BINARY SEARCH MAX.  7 BYTEs
. filename [argu1] [argu2]...<RETURN> : SHELL
?                        <RETURN> : HELP
Q                        <RETURN> : QUIT
===============================================
* The information listed below is for reference only :
  The absolute start address of BUFFER : 2E24:0000
  The Buffer size : 256 K BYTES


==
```

Edit command summary

The "==" symbol is the command prompt for the editor. Use the "?" command at the prompt to display the available editor commands at any time.

If the PC memory is assigned as the working buffer then the actual buffer base address in the PC memory is shown at the bottom of the command summary. It is the real base address of the buffer in the PC memory and is displayed in SEGMENT : OFFSET form.

For special purposes, some advanced users pass this address to their own editing program which can edit the buffer contents in different ways.

EXAMPLE: When the fictitious editor, FRED.EXE, is applied, the user can use the DOT command to pass the base address as follows:

. FRED SEGMENT : OFFSET < CR >

This will work for editors that have the command line capability to take the PC memory segment and offset parameters as inputs. For example:

A>FRED SEGMENT:OFFSET < CR >

You may return to the main menu by typing Q and < CR >.

4.5 Display Buffer

Press 7 to select the display buffer function. This function will display the working buffer contents ranging from the BUFFER START ADDRESS to the BUFFER END ADDRESS as shown in the status field. Data will be displayed on the screen, in HEX and ASCII format as follows:

```
Start   End          File name
00000---00000 :
00000---00000 :
00000---00000 :
00000---00000 :
00000---00000 :
00000---00000 :
00000---00000 :
00000---00000 :
00000---00000 :
00000---00000 :
00000---00000 :
00000---00000 :
00000---00000 :
00000---00000 :
00000---00000 :
00000---00000 :
00000---00000 :
00000---00000 :
00000---00000 :
00000---00000 :

Press any key to continue ....
```

Display buffer

Pressing < ctrl > + "S" can hold the display. Pressing < ESC > will terminate the display. After terminating the display, press any key to return to the main menu.

4.6 Change I/O Base Address

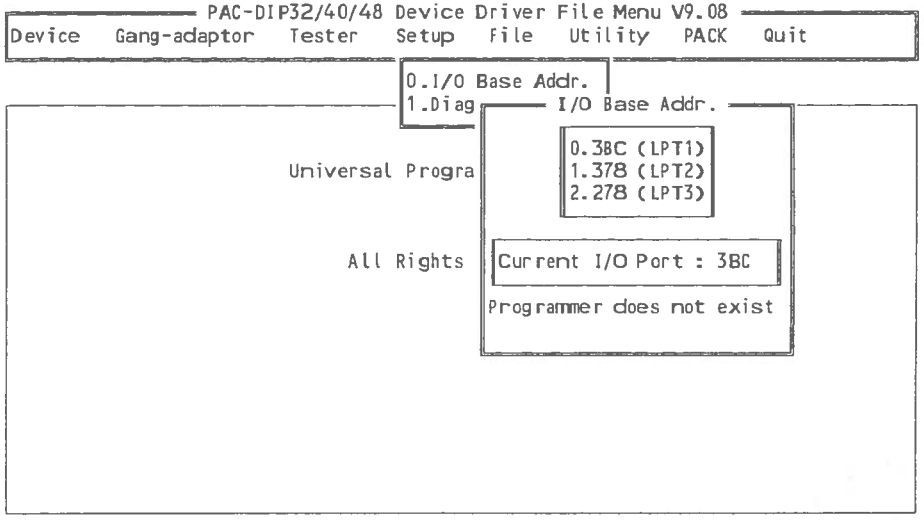
Before performing any functions the device driver file's I/O address setting must be set to match the address of the parallel/printer port the Universal Programmer is connected to or to match the address set on the internal interface card being used.

When using the internal interface card you select the I/O address by positioning the jumper of the interface card. The computer power must be switched off when changing this jumper position. The default I/O address of the internal

interface card supplied with the Universal programmer is 278 (LPT3). The default address for the parallel/printer port on most PC's is 3BC or 378 (LPT1). The I/O address jumper of the interface card has following three choices:

3BCH (LPT1)
378H (LPT2)
278H (LPT3)

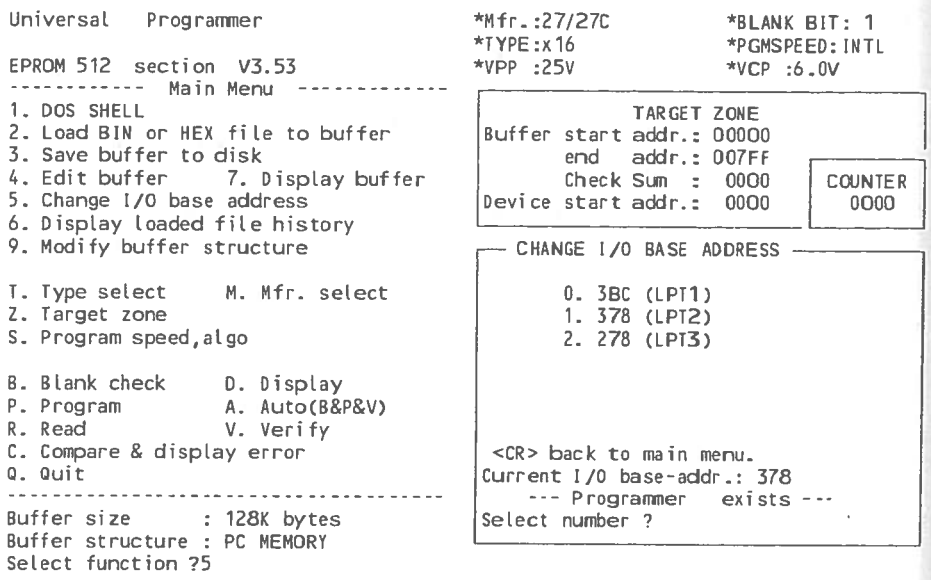
If you want to change the device driver's I/O address under the ACCESS Menu press letter "S" and number "0" to obtain the following menu:



Then select the I/O address that corresponds to the address of the hardware port the Universal programmer is connected to. The software will automatically check the hardware I/O circuits and display the result "Programmer Exists" if everything is set up OK. If the "Programmer Does Not Exist" message appears then the software I/O address needs be changed to match the hardware I/O address. It is necessary to check both I/O addresses carefully.

When "Programmer Exists" appears, press " * " to return to the ACCESS Menu, and then press Q. Now ACCESS will write the software I/O address value to the ACCESS.DAT file as reference for the next ACCESS execution.

Each time a driver file is executed the software first checks that the software and hardware are set to the same I/O base address and communication is functioning properly. If the address set in software does not match the address set in hardware then the message "Error Identification on Hardware -Press Q to quit or < CR > to continue" is displayed. Press < CR > to enter the device driver file main menu. Press "5" to select the "Change I/O base address" function. A dialog window will be displayed on the screen as follows:



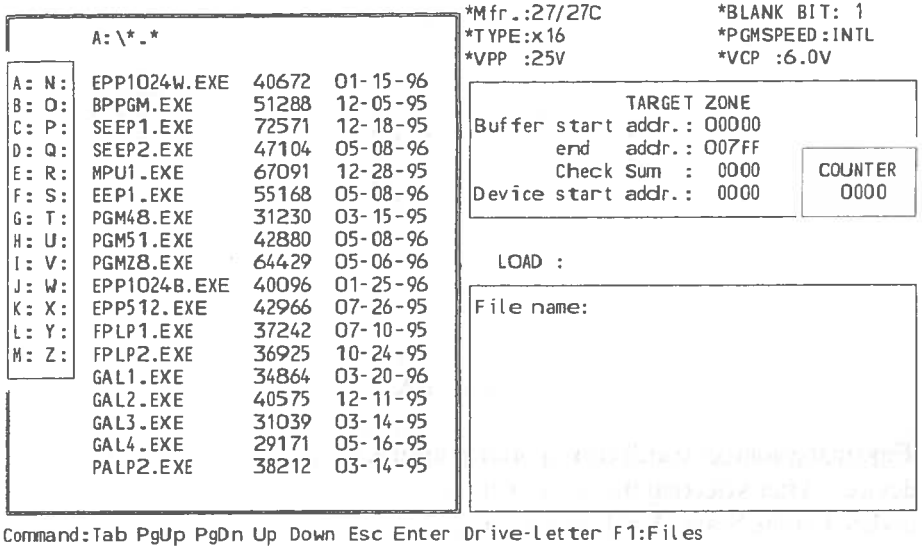
When prompted, type in the number of the new I/O address you would like to use. The I/O address you have selected will be updated in the internal buffer for programming reference.

Press < ESC > or < CR > to return to the main menu. When you press "Q" to

QUIT the driver file will save the new I/O address in the parameter file ????????.DAT for later reference. The I/O base address only needs to be set one time and then will be remembered by the software.

4.7 Display Loaded File History

Press "6" to select the list file information function. The last 20 loaded files and their loaded addresses will be displayed on the screen for reference.



List file information

Press <ESC> to return to the main menu.

4.8 Manufacturer, Prefix Select

Press M, and the submenu will be displayed as shown.

```
Universal   Programmer                  *Mfr.:27/27C          *BLANK BIT: 1
                                        *TYPE:x16             *PGMSPEED:INTL
EPROM 512  section  V3.53               *VPP :25V             *VCP :6.0V
----------  Main Menu  ------------
1. DOS SHELL                                  TARGET ZONE
2. Load BIN or HEX file to buffer     Buffer start addr.: 00000
3. Save buffer to disk                       end   addr.: 007FF
4. Edit buffer      7. Display buffer        Check Sum  :  0000     COUNTER
5. Change I/O base address            Device start addr.:  0000       0000
6. Display loaded file history
        Mfr. SELECT :
00. 27/27C          09. MATSUSHITA      18. PHILIPS         27. SONY
01. AMD/MMI         10. MICROCHIP       19. SMOS            28. ICT
02. ATMEL           11. MITSUBISHI      20. SGS_THOMSON     29. OMNI-WAVE
03. CATALYST        12. MOSTEK          21. TI              30. SHARP
04. CYPRESS         13. MOTOROLA        22. TOSHIBA         31. Winbond
05. FUJITSU         14. NEC             23. WSI             32. ISSI
06. HITACHI         15. NS              24. VLSI
07. HYUNDAI         16. OKI             25. SEEQ
08. INTEL           17. RICOH           26. MXIC

   <CR> back to main menu.
SELECT NUMBER ?
Select function ?M
```

Submenu of Mfr.

Type in the number which corresponds to the intended manufacturer of the target device. After selecting the manufacturer, the new manufacturer status will be updated in the Status Field for reference.

4.9 Type Select

Press T, and the submenu will be displayed as follows:

```
Universal   Programmer                  *Mfr.:27/27C          *BLANK BIT: 1
                                        *TYPE:x16             *PGMSPEED:INTL
EPROM 512  section  V3.53               *VPP :25V             *VCP :6.0V
----------  Main Menu  ------------
1. DOS SHELL                                  TARGET ZONE
2. Load BIN or HEX file to buffer     Buffer start addr.: 00000
3. Save buffer to disk                       end   addr.: 007FF
4. Edit buffer      7. Display buffer        Check Sum  :  0000     COUNTER
5. Change I/O base address            Device start addr.:  0000       0000
6. Display loaded file history
        TYPE SELECT :
0.x16  25V                          9.x256  12.75V
1.x16B  12.5V                       10.x256HV  21V
2.x32  25V                          11.x512  12.75V
3.x32A  21V                         12.x512HV  21V
4.x32B  12.5V                       13.x513  12.75V
5.x64  21V                          14.SRAM 6116
6.x64A  12.5V
7.x128  21V
8.x128A  12.5V

  <CR> back to main menu.
SELECT NUMBER ?
Select function ?T
```

Submenu of Type

Type in the number or letter that precedes the type listing of your target device. The type and voltage required for that device will be displayed in the STATUS FIELD for reference.

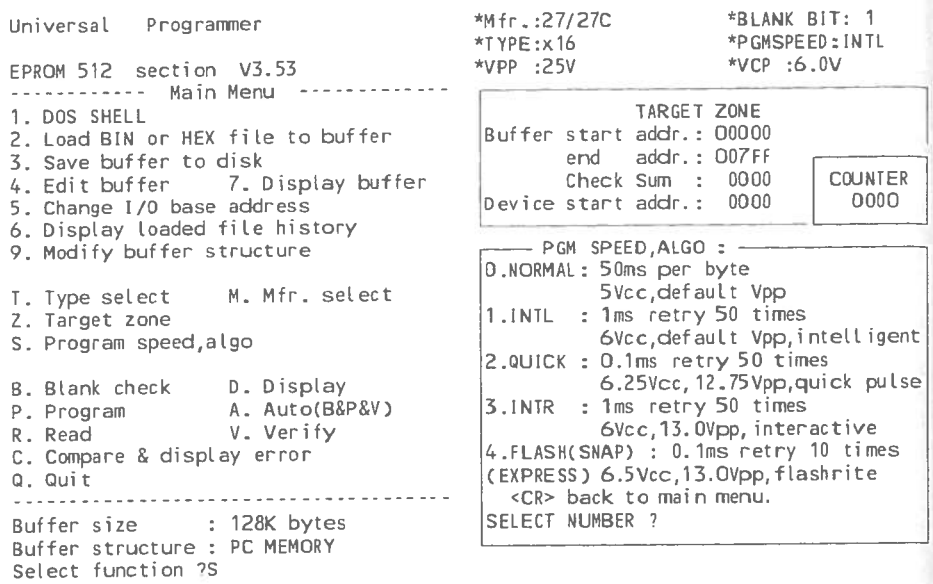
Press < ESC > or < CR > to return to the main menu.

NOTE

EPROM chips are not always clearly labeled; it may be difficult, for example, to determine whether the EPROM you wish to program is a 2732 requiring 25 V, or whether it is a 2732A, which requires 21 V. If you encounter this problem, try to program the EPROM at the lower voltage first; if this does not work, then erase the EPROM and try the higher voltage.

4.10 Program Speed, Algorithm

Press S to select the speed function. A dialog window will be displayed on the screen as follows:



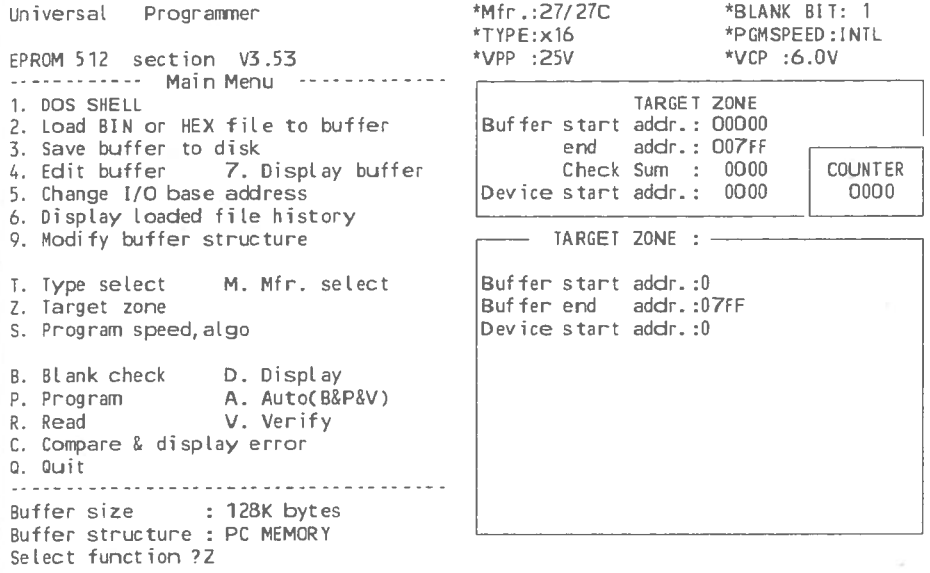
Speed select

When prompted, type in the number of the new programming speed desired. The speed selected will be updated in the STATUS FIELD for programming reference. This selection is only available for 27Cxxx series EPROM devices. Most newer devices can use the QUICK pulse algorithm. Some EPROM manufacturers require a proprietary algorithm which should not be changed from the default.

Press < ESC > or < CR > to return to the main menu.

4.11 Modify Buffer Range (Target Zone)

Press Z to select the modify buffer function. This function is used to select what portion of the buffer is programmed to a target device and what portion of the buffer is included in the checksum calculation. Usually this function is used to limit the size of the buffer if a file is small compared to the EPROM size of the target device you will program. A dialog window will appear on screen as follows:



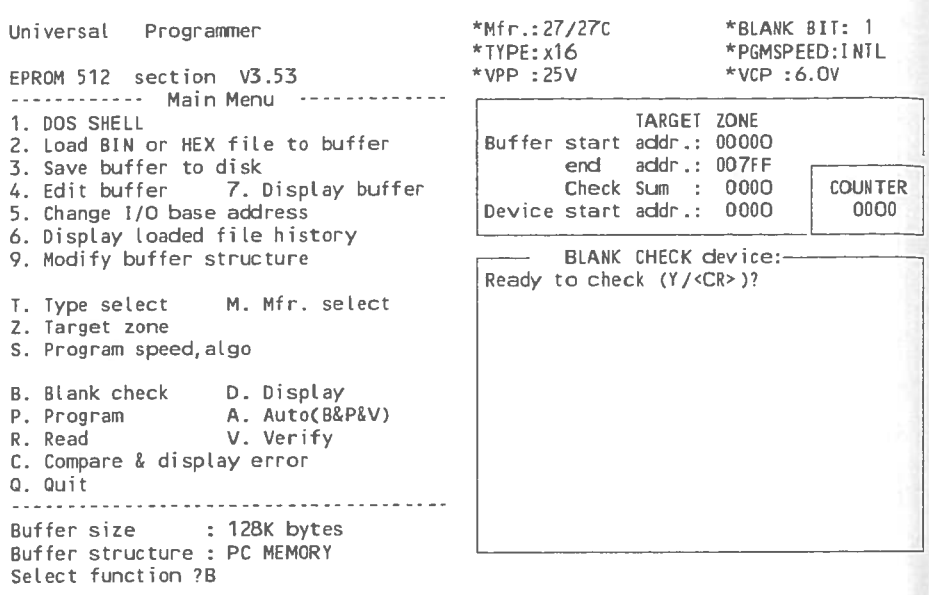
Modify buffer

When prompted, type in the new buffer start address and press < CR >. Next, type in the new buffer end address and press < CR >. Then, type in the new EPROM start address and press < CR >. The EPROM start address is the address in the target device where the information starting at the buffer start address will be programmed to in the target device. The values entered in HEX code will be updated in the STATUS FIELD for reference. But, when the type or manufacturer (mfr) selection is changed the target zone will default to the normal target zone specified by the device selected. The normal address range is equal to the entire address range of the target device.

Press any key to return to the main menu.

4.12 Blank Check

Press B, and the Blank check window will be displayed as shown.

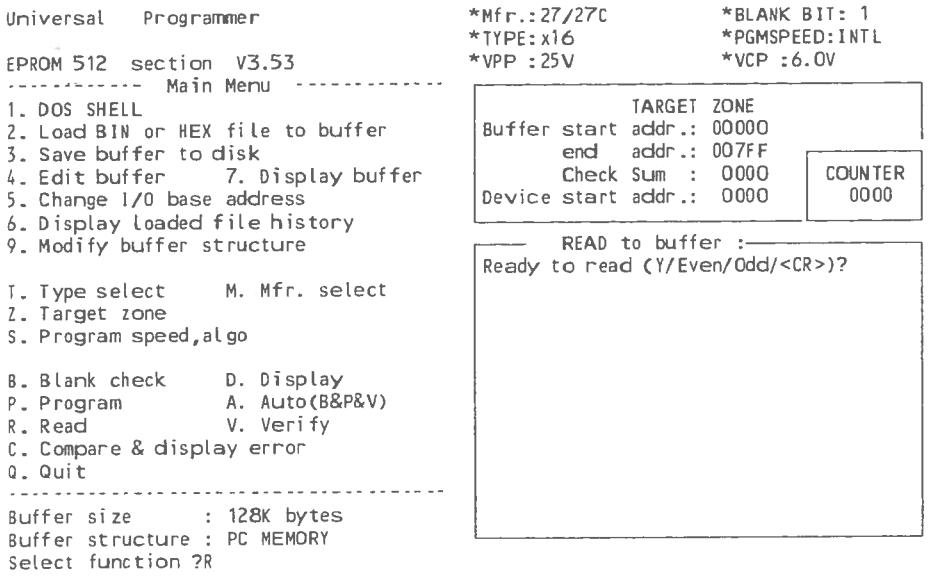


Submenu of Blank check

The blank check function is used to check whether a target device is ready to be programmed. All devices except for EEPROM devices should be in the blank state before being programmed. Press "Y" on the keyboard or the < YES > key on the Universal programmer to start the blank check. If the device fails the test, the first address that is not blank will be displayed. Otherwise, the "Blank Check OK !" message will be displayed.

4.13 Read

Press R, and the Read window will be displayed as shown.



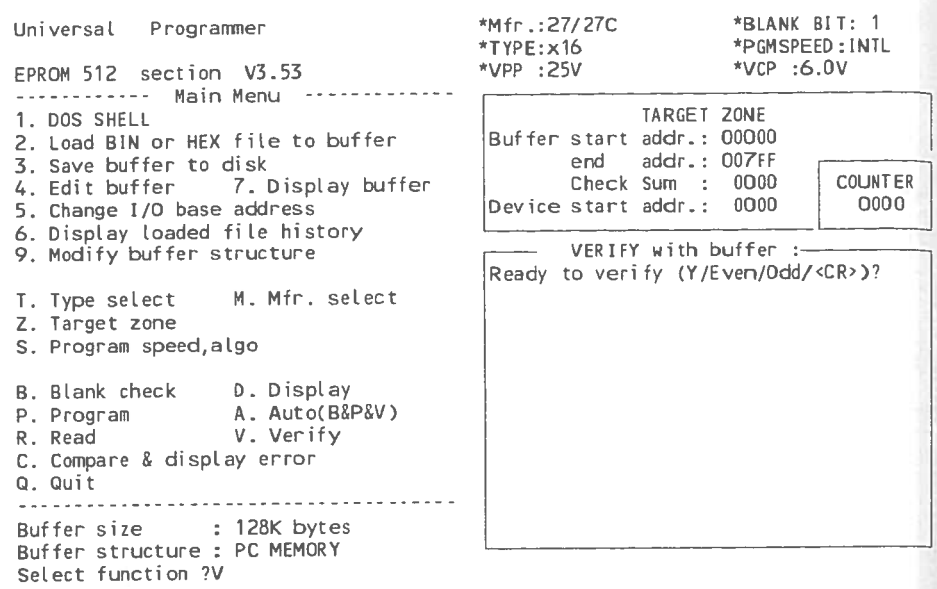
Submenu of Read

Press " Y " to read data from the EPROM to the buffer shown in the STATUS FIELD, or press < CR > to return to the main menu. The submenu will display the Reading now... message, and when the reading is completed, the "Read OK!" message will be displayed.

The check sum of all the data read into the buffer will be automatically calculated after the reading a device.

4.14 Verify

Press V, and the verify window will be displayed as shown.



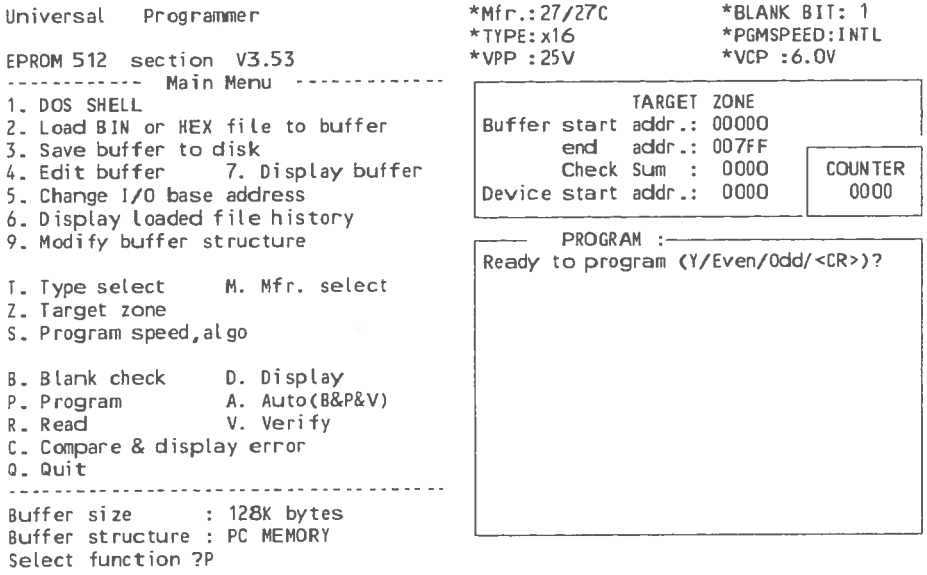
Submenu of Verify

Press "Y" to verify that the target device contents match the contents located in the buffer at the address range shown in the STATUS FIELD, or press < CR > to return to the main menu. The submenu will display the "Verifying now" message. When the verify function has been completed, the Verify OK! message will be displayed on the screen.

The verifying routine will be terminated and an error message will be displayed if the contents of the device under test do not match the contents found in the working buffer.

4.15 Program

Press "P" and the program window will be displayed as shown.

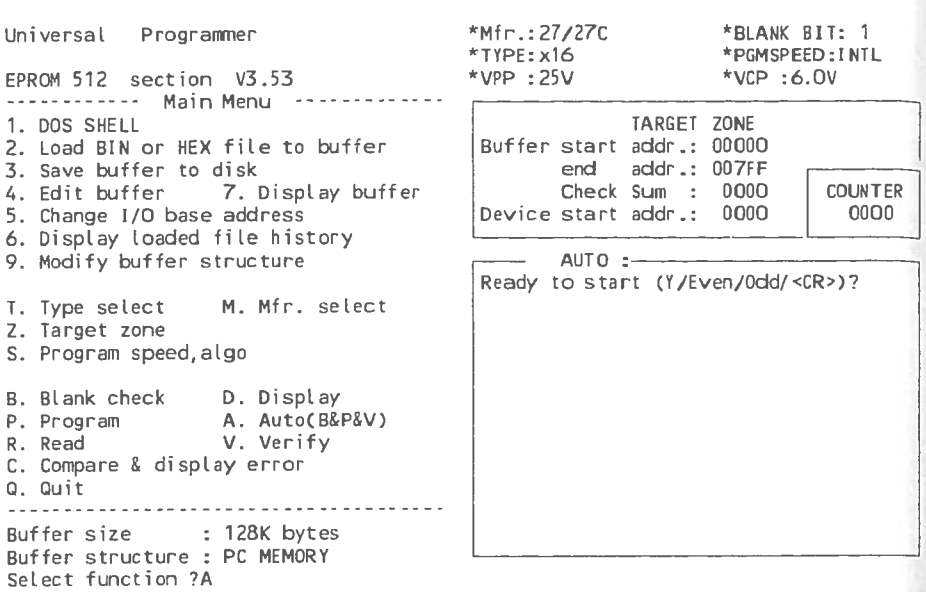


Submenu of Program

Select "Y" to begin transferring the data in the working buffer to the target device, or press < CR > to return to the main menu. The programming message will appear on the screen and the counter in the Status field will increment until the function is completed. Then, a Program OK ! message will be displayed on the screen. Immediately following the "Program OK !" message the device will be verified against the buffer contents as a final check that the device has been programmed properly.

4.16 Auto (B & P & V)

Press "A" and the Auto window will be displayed as shown.

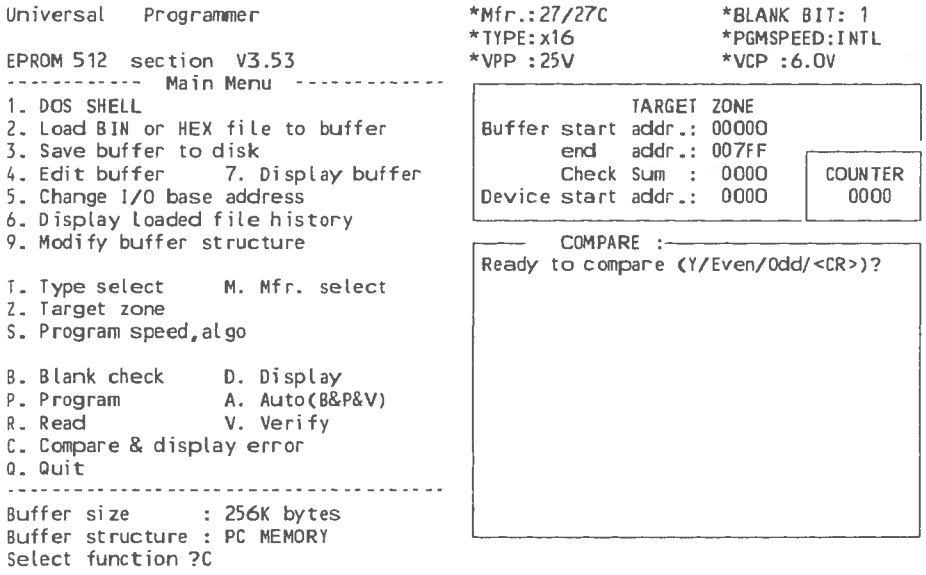


Submenu of Auto

You can then press "Y" to start the Auto function, or press < CR > to return to the main menu. This function is similar to the Program function, but it also automatically performs a check before programming and the security fuse is automatically blown after verifying. Please note that not all device types have security fuses.

4.17 Compare & Display Error

Press C, and the compare window will be displayed as shown.



Submenu of Compare

This function will compare the buffer shown in the Status Field with the device in the socket(s) and display any differences on the screen.

Press "Y" if you wish to compare the data on the device with the working buffer, and the screen will display the differences in the format "address:data" where the buffer contents are enclosed in parenthesis. Otherwise, press < CR > to return to the main menu.

Error at :
Press < ESC > to terminate display.
00000:DA - (00000:21) , 00001:1F - (00001:05),
Press any key to continue.

While the differences are being displayed, you can press < ESC > to terminate the display, then press any key to return to the main menu. If no errors are found, the Compare OK! message will be displayed.

4.18 Display

Press D, and the display window will be displayed as shown.

```
0680  00 0D 5F 63 6F 6D 70 61 --72 65 5F 6D 61 6E 75 00 .._compare_manu.
0690  07 5F 66 63 6C 6F 73 65 --00 0D 5F 70 72 6F 67 72 ._fclose.._progr
06A0  61 6D 5F 6D 61 6E 75 00 --07 5F 68 6C 69 67 68 74 am_manu.._hlight
06B0  00 0E 5F 63 68 65 63 68 --5F 6D 61 63 68 69 6E 65 .._check_machine
06C0  00 0A 5F 61 75 74 6F 5F --6D 61 6E 75 00 07 5F 63 .._auto_manu._c
06D0  75 72 5F 6F 6E 00 09 5F --63 68 6B 5F 36 38 34 35 ur_on.._chk_6845
06E0  00 07 5F 73 74 72 63 70 --79 00 06 5F 68 74 6F 75 .._strcpy._htou
06F0  6C 00 08 5F 73 76 31 33 --6C 6F 77 00 08 5F 67 65 l._sv13low._ge
0700  74 6C 69 6E 65 00 08 5F --63 68 64 72 69 76 65 00 tline._chdrive.
0710  07 5F 73 76 31 33 68 69 --00 61 B0 07 00 02 5F 63 ._sv13hi.a...._c
0720  00 62 02 21 8C 12 00 06 --5F 63 68 64 69 72 00 07 .b.!...._chdir..
0730  5F 63 6C 73 63 72 6E 00 --08 B0 0D 00 08 5F 76 69 _clscrn......_vi
0740  64 5F 6D 65 6D 00 62 04 --95 8C 81 00 05 5F 65 78 d_mem.b......_ex
0750  69 74 00 07 5F 77 61 69 --74 6D 73 00 0B 5F 64 65 it._waitms._de
0760  62 75 67 5F 6D 61 6E 75 --00 07 5F 73 74 72 6C 65 bug_manu._strle
0770  6E 00 08 5F 65 72 72 62 --65 65 70 00 0A 5F 6C 6F n._errbeep._lo
0780  61 64 5F 6D 61 6E 75 00 --0A 5F 73 61 76 65 5F 6D ad_manu._save_m
0790  61 6E 75 00 07 5F 73 70 --61 77 6E 6C 00 0A 5F 68 anu._spawnl._h
07A0  69 73 74 5F 6D 61 6E 75 --00 0B 5F 64 69 73 70 6C ist_manu._displ
07B0  61 79 5F 63 68 00 0C 5F --64 73 70 62 75 66 5F 6D ay_ch._dspbuf_m
07C0  61 6E 75 00 06 5F 66 6F --70 65 6E 00 9C B0 0C 00 anu._fopen.....
07D0  07 5F 76 70 70 5F 6E 6F --00 62 02 E8 8C 2B 00 08 ._vpp_no.b...+..
07E0  5F 66 70 72 69 6E 74 66 --00 0C 5F 64 73 70 6D 61 _fprintf._dspma
07F0  73 5F 6D 61 6E 75 00 07 --5F 69 6E 69 74 69 6F 00 s_manu._initio.
Press any key to continue
```

Submenu of display

The screen will display the addresses and the current contents of those addresses in the target device in the socket. For PAC's with multiple sockets this function displays only the contents of the device in socket number 1. Press < ESC > to exit the display, and press any key to return to the main menu.

4.19 Modify Buffer Structure

The Universal Programmer software only uses the conventional memory of a PC which allows a buffer space of no more than 256K bytes. This is about the size of a buffer required for a 2Mbit EPROM. The modify buffer structure is

used to resolve this limitation. The modify buffer structure function is used to reassign the buffer to drive C:, D:, E:, or F:. When the hard disk is used as the working buffer programming delays can occur due to the limit of the hard disk access time. These delays can be overcome by using RAMDRIVE.SYS to form RAM disk E: or F: which can then be assigned as the working buffer.

After the Quit function the parameters of the modify buffer structure will be saved to ????????.DAT. It will be automatically recalled when the driver file is again used.

4.20 Security, Lock Bits & Encryption Code

Some MPUs, in addition to using ROM code programming, have other facilities such as Lock Bits, Security Bits or Encryption Code to prevent unauthorized copying of proprietary code. The user can easily operate these functions by selecting "L", "S" or "E" under the main menu, or use "Auto" function to program the Lock Bits, Security Bits or Encryption Code (Refer to each individual MPU's manual for the definition of these special options).

4.21 Quit

Press "Q" to exit the main menu and to return to DOS. If you are in one of the sub-menus, you must first exit that submenu and return to the main menu before the "Q" function will work.

Before exiting to DOS, the software will save parameters including manufacturer, type, I/O address and programming speed in the ????????.DAT file associated with driver file in use.

4.22 Batch File Function

The batch file function is an easy, time saving, and accurate method to operate the Universal Programmer often used for mass production setup. Batch file operation is accomplished using DOS's redirected input function. Keystrokes are stored as a DOS text file and passed directly to a Universal Programmer driver file.

The following is an illustration of how to operate the Universal Programmer in batch mode using the AMD 27C512 as an example.

1. The normal operation with EPP512.EXE under DOS requires following steps:

<M>	selects Mfr. function
<01> <enter>	selects AMD as the manufacturer
<enter>	brings you back to the main menu
<T>	executes the Type select function
<B> <enter>	chooses 27C512 as the type
<2> <test.bin> <enter>	loads file "test.bin" to the buffer
<B> <0> <enter>	selects binary file type with load address 0000
<enter>	returns to main menu
<B> <Y> <enter>	selects and executes blank check function
<P> <Y> <enter>	selects and executes program function

2. The complete batch file to run these commands should be written as follows:

```
M01 <enter>
<enter>
tb <enter>
2test.bin <enter>
<enter>
by <enter>
py <enter>
```


A batch file for the Universal programmer can be edited by a standard text editor like PE3,... To use this batch file with the EPP512.EXE driver file type in the following at the DOS prompt:

```
C:\PT>EPP512<test.doc
```

The batch file in this illustration will select the Program function and start programming a device. If you leave out the last "y" and < enter > key the batch file will prompt the user "Ready to Program?" and wait for the "Y" or "YES" key to be pressed to program the device. This allows 1 key operation for production programming.

Note:

> Some text editors add an extra carriage return at the end of the last line when you save the program. This may cause unwanted results like the last function be entered and then returning to the main menu without the operator being prompted to start the operation.

4.23 External Key, LEDs and Side Power Switch

The only external key on the programmer module is the "YES" key. It functions exactly the same as the "Y" key on the PC keyboard. When the submenu asks you to enter "Y", you may press the "YES" key instead of pressing the "Y" key on the PC keyboard.

The three LEDs on the programmer module are labeled ON , BUSY, and GOOD.

The ON LED will light up after turning on both the computer power switch and the programmer's side power switch. The BUSY LED will light up during device programming, reading and verifying. The GOOD LED will light up when the result of an operation is positive.

NOTE

1. The programmer's side power switch must absolutely be turned on before running the ACCESS file on the hard disk or device driver file on floppy disk.
2. When the programmer is not being used, the user can switch off the side power switch which will turn off the +5V, +12V, -12V power supplied to the programmer which can help to prolong the life of the equipment..

About PLD (PAL, GAL, PEEL, EPLD)

The functions used for PLD programming are almost the same as those used to program EPROMs. The following section describes the additional functions.

4.24 Additional or Alternate Functions for PLD Programming

Load JEDEC File:
To program a PLD the user needs to load a JEDEC Fuse Map file instead of a Binary file. This is a standard file format produced by nearly all logic compilers such as ABEL, CUPL, PALASM II, SLICE, PDK-1, PLAN II. Some PLD/CPLD's require proprietary formats which are also supported by the device driver files where appropriate.

Save JEDEC File:
Under PLD Device Driver file, data is saved in JEDEC file format instead of binary format.

Edit Buffer:
Though the Edit Buffer function is included, we recommend you use it onlyfor viewing. To edit the buffer directly, the user must first master the PLD JEDEC Fuse Map assignment. We advise to only edit with a PLD compiler.

Display Buffer, Display Device:
Under PLD Device File, both display functions display files in the JEDEC format.

Security Fuse Blow:
Pressing "S" to blow the security fuses is the final step in PLD programming. Blowing the security fuses can prevent any further access to the PLD either for "modification" or for "reading". Blowing the fuses prevents anyone from making unauthorized copies of your PLD. Please note that the "Auto" function will automatically blow the security fuse. After the security fuse is blown the device can not be read or verified.

5. UTILITIES

5.1 HEX TO BINARY code converters

Any data loaded to the working buffer must be first converted to binary format to be accepted by the driver file. These HEX converters will convert your HEX format files to executable BIN code that is recognizable to the target CPU. Even though the programmer can load HEX files directly, some users may like to use this converter to convert all their HEX files to binary format so that files will be downloaded at a faster rate to the working buffer. If, for example, a user has a 2 Mbit or larger file that needs to be split into several files, these HEX to BIN converters will be useful to produce files that other utilities can work on. (SPLITX.EXE and SHUFFX.EXE accept only binary files as input.)

There are two HEX file converters: HEXBIN and HEXBIN2.HEXBIN.EXE, with a maximum conversion size of 64K bytes, can be used to convert the following HEX formats to binary (BIN) format.

INTEL HEX format
MOTOROLA S HEX formats
TEKTRONIX HEX format
TI SOSMAC format
STRAIGHT HEX format
MD16 HEX format

HEXBIN2.EXE, with a maximum conversion size of 4096K bytes, can be used to convert the following HEX formats into binary (BIN) format.

(I) INTEL HEX
(M) MOTOROLA S HEX (S record)
(T) TEKTRONIX HEX
(D) DIGITAL RESEARCH HEX
(H) INTEL HEX-32 extended HEX

HEXBIN2.EXE allows specifying a starting address, and the leading "garbage" of the file will be skipped to minimize the output binary file size. This is the main difference between HEXBIN2.EXE and HEXBIN.EXE.

The HEX to BIN converters can be run in the command line mode or in the prompt mode. In the command line mode the converters can be operated as follows at the DOS prompt:

```
> HEXBIN2.EXE  [HEX file name] [BIN file name] [HEX format]
               [start address] [end address] [unused byte selection]
```

All entries in brackets are optional.

HEX file name and BIN file name are the standard file names specified in DOS.

The HEX file name is the input file and the BIN file name is the output binary file to be produced by the utility. Start address and end address are Hexadecimal digits.

INTEL extended HEX
In the INTEL extended HEX format, the start address represents the start segment address. All data ranging from the start segment X 16 to the end of the file will be converted to binary format.

MOTOROLA S HEX format
In the MOTOROLA S HEX format the start address represents the actual file start address. All data ranging from the start address to the end of the file will be converted to binary format.

Unused byte select: To fill the unused bytes with 00 or FFH. Use 1 to fill with "00" and 2 to fill with "FF".

For example:

```
A:> HEXBIN2 DEMO.HEX DEMO.BIN I 1000 FFFF 1 < CR >
```

HEXBIN2 will convert the HEX file DEMO.HEX to binary file DEMO.BIN

through INTEL HEX converting technique, and the data ranging from 0000H to the end of the data will be converted.

These programs can be used in the prompt mode as follows:

```
A:> HEXBIN2 < CR >

HEX FILE NAME :   DEMO.HEX
BIN FILE NAME :   DEMO.BIN
HEX FILE FORMAT < I >NTEL  < M >OTOROLA < D >IGITAL RESEARCH
                < T >EKTRONIX  < H >INTEL HEX-32
Input code segment start address        [0000]
Input code segment end address          [FFFF]
Unused bytes will be < 1 >00 < 2 >FF [ I ]
```

5.2 Dump Binary File To Console

DUMP.EXE can convert an executable BINARY file to HEXADECIMAL characters to be displayed on screen or sent to a printer. The DOS type command can not display binary files.

The input format under the DOS command prompt is:

```
A:> [^P] DUMP.EXE FILENAME [start address] < enter >
```

All entries in brackets [ ] are options.

^P : <CTRL> + P
The ^P option can be used to send the data in the file a printer. The data will then be displayed on screen and will also be output to the printer.

FILENAME is a standard filename as specified in DOS.
Start address is a hexadecimal digit. All data from the start address to the end of the file will be displayed.

5.3 2-way, 3-way, 4-way, 8-way, 2-way word Binary File Splitter

SPLIT2.EXE can split a 16-bit binary source file into two 8-bit binary files. One output file contains the collection of bytes stored in the odd bytes of the original source file starting with the byte at address 1,3,5,... The other output file contains the collection of bytes stored in the even bytes of the original source file starting with the bytes stored at address 2, 4, 6,....). This utility is used to split a file for systems that have one 16 bit input file that will be stored in 2 8 bit EPROM/FLASH to form a 16 bit bus.

SPLIT3.EXE,
splits a source file of 24-bit into three 8-bit files to be programmed in sequence.

SPLIT4.EXE,
splits a source file of 32-bit into four 8-bit files to be programmed in sequence.

SPLIT8.EXE,
splits a source file of 64-bit into eight 8-bit files to be programmed in sequence.

SPLIT216.EXE,
splits a source of 32-bit into two 16-bit files to be programmed in sequence.

The input command under the DOS command prompt is:

```
A:>SPLIT2  [input file] [output EVEN file] [output ODD file name] < enter >
A:>SPLIT4  [inputfile] [output 1st file] [output 2nd file]
           [output 3rd file] [output 4th file]<enter>
```

All items in brackets[ ] are standard filenames specified under DOS.

5.4 2-way, 4-way, Binary File Shuffler

SHUFF2.EXE can shuffle two 8-bit binary source files to become a single 16-

bitbinary file. The first byte of first 8-bit file will be located at the 1st byte of the 16 bit output file. The first byte of the second 8-bit file will be located at the 2nd byte of the 16-bit output file.

SHUFF4.EXE can shuffle four 8-bit binary format source files to become a single 32-bit binary file. The first byte of the first 8-bit file will be at the 1st byte of the 32 bit output file. The first byte of the second 8-bit file will be at the 2nd byte of the 32 bit output file. The first byte of the third 8-bit file will be at the 3rd byte of the 32 bit output file. The first byte of the fourth 8-bit file will be at the 4th byte of the 32 bit output file.

The input command under the DOS command prompt is :

```
A:>SHUFF2  [output file] [input EVEN file] [input ODD file] < CR >
A:>SHUFF4  [output file] [input 1st file] [input 2nd file]
           [input 3rd file] [input 4th file]
```

Again, all terms in brackets are standard DOS file names.

5.5 Binary to Hex Converter

BIN2HEX.EXE can convert BIN file to INTEL or MOTOROLA S-HEX file format, the command format is as follows:

```
A:>BIN2HEX  [input BIN file] [output HEX file] [HEX format]
            [input BIN file start address] [output HEX file start address]
            ([S-HEX format data length])
```

Input BIN file & Output HEX file are the complete DOS type file name. HEX format supported are INTEL HEX, MOTOROLA S1, MOTOROLA S2 and MOTOROLA S3.

The input BIN file start address asks for the first address in the binary file that

should be converted and placed in the HEX file. All data from this address to the end of the file will be converted. When INTEL HEX format is selected, the input address will be the segment address (real address = segment address x 16)

Output HEX file start address asks for the offset address in the HEX file where the BINARY data should be loaded to.

When INTEL HEX is entered, the input address asks for segment address (real address = segment address x 16)

For example:

```
BIN2HEX  TEST.BIN TEST.HEX  I 0001  0000
```

if BINARY file content is as follow

```
0000 00 01 02 03 04 05.....
0010 10 11 12 13 14 15.....
```

the HEX file result will be

```
(0000 10 11 12 13.....)
:10 0000 00 10 11 12 13.....
```

the input command under DOS prompt is:

```
A:> BIN2HEX <CR>
A:> Input BIN file name: TEST.BIN
A:> Output HEX file name : [TEST.HEX]
A:> Select HEX format ( 0:INTEL, 1:MOTOROLA S1, 2:MOTOROLA S2,
                        3:MOTOROLA S3 ) [0]:
A:> Input BIN file start segment     [0000]
A:> Output HEX file start segment  [0000]
Select S-HEX format data length (0:16, 1:32, 2:64) [1]:
```

Only when Motorola S-HEX is entered will the above shadowed line appear for the user to modify the data length of each line.

APPENDIX A.

IC TESTER

The programmer's circuits also include an IC tester function for testing following 5 IC types:

1. 74/54 TTL, HC, HCT or equivalent CMOS series
2. 40/140 CMOS series
3. 45/140 CMOS series
4. SRAM 6116, 6264, 62256, 68128(1M), 58400(4M) series
5. DRAM 4164, 41256, 411000, 4416, 4464, 44256, 44100(4Mx1), 414000(1Mx4).

Available Functions:

FUNCTION TEST
LOOP-TEST
AUTO-SEARCH NUMBER
USER DEFINED VECTOR TESTS

File Contents of the Software Disk

ICTEST.EXE:	Main program file.
ICTEST.DAT:	Setup and parameter data file.
README.DOC:	Records of any revision or amendments to the IC test card or software since the printing of manual is released.
TTL74.LIB:	TTL library.
CMOS40.LIB:	CMOS 40 library.
CMOS45.LIB:	CMOS 45 library.
7406.VEC:	Sample vector file.
4040.VEC:	Sample vector file.

1 Basic Operation

There are two ways to operate the IC Tester.

1.1 Enter "T" from the ACCESS menu, then select the intended IC Group to test.

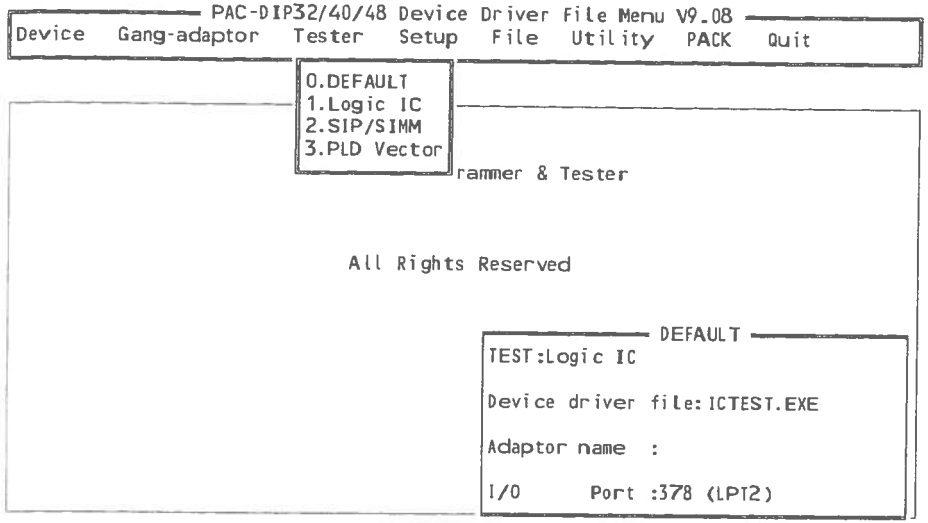


Figure of ACCESS

Press "1" for the "Logic IC" tester item, the screen will display the main menu as follows:

```
Universal   Programmer                    * TYPE: TTL74      * NUMBER: 00

ICTEST  section  V3.37

Main Menu:

  1. DOS SHELL
  2. Change I/O base-addr.
     of hardware

  T. IC Type select
  N. IC Number specify
  F. Function test
  L. Loop test
  S. Search unknown IC number
  U. User defined test vector

  A. DRAM test program
  B. SRAM test program
  Q. Quit

  Select function ?
```

main menu

1.2. Alternatively the ICTEST.EXE file can be run directly from the DOS prompt to enter the above main menu.

2 Function Guide

To select a particular function enter the number or that letter precedes that function.

1. DOS SHELL

Press 1 to select the DOS SHELL function. After entering into this function, the software will look for the file COMMAND.COM at the DOS boot disk drive. If it exists, the software will execute this file and pass the control over to a DOS command prompt as follows:

```
Use EXIT to return to Programmer
    Microsoft (R) MS-DOS (R)  Version 3.30
                (C) Copyright Microsoft Corp 1981-1987

    C: \PT\ICTEST>>
```

The software is now controlled by DOS and waiting for your command. The command format is the same as the DOS command format. To return to the main menu just type "exit" and press <enter> under the DOS prompt.

NOTE

This function will invoke the COMMAND.COM. The user has to prepare this file on the DOS boot disk drive, otherwise the function will not work.

2. Change I/O Base address
This function operates the same in all driver files, please refer to Section 4.6 for more details.

T. IC Type selection:
There are three IC types to be selected, TTL74, CMOS40 and CMOS45. Press the desired number and the corresponding library will be selected for testing.

N. IC Number specify:
Once a library is selected enter only the serial number of the IC to be tested for this function, e.g., 174 instead of TTL74174. In this case the TTL74 prefix was already selected by the IC type (T) function. The number entered will be checked to see if it is in the standard library. If the number is not available you can write user defined test vectors with the U function (described below) to test the device. See function U below.

F. Function test:
This is a single loop function test, i.e., the possible logic combinations are carried out only once.

L. Loop test:
This option will repeatedly test the chip logic combinations in an endless loop. In the event of an error during any of the tests, the testing will stop and an error message will be shown. The test may be terminated by pressing any key.

S. Search unknown IC number:
If you have an IC with an illegible type marking the search function will apply vectors to the device to determine what device type is in the socket. If the devices response does not match any of the devices in the standard vector library included with the Universal Programmer software then a message saying "IC not found" will be displayed. The parameters of an unknown chip may be contained in any of the three libraries included (TTL, CMOS 40, CMOS 45). It may be necessary to search through all three libraries in turn to find out what device you have.

U. User defined test vector:
This function allows any logic input, expected output, 5V and GND to be defined on the 24 pins of the test socket.

You may write your own vectors to test any sequence of logic steps for new IC (including PAL, EPLD, PROM etc.). The test vector syntax is described in detail in Appendix A. point 4 on page 64. (NOTE: Only logic testing is carried out. Loading and IC speed tests are excluded.)

A. DRAM test program:
When the function is entered, you will have following three choices:

N. DRAM number select:	List all available types for selection.
F. DRAM function test:	Enables all bits on the DRAM to be written and read back for comparison.
Q. Quit	Quit to IC test main menu.

B. SRAM test program:
When the function is entered, you will have following three choices:

N. SRAM number select:	List all available types for selection.
F. SRAM function test:	Enables all bits on the SRAM to be written and read back for comparison.
Q. Quit	Quit to IC test main menu.

Q. QUIT:
QUIT to DOS. The current I/O Base Address and the current IC type will automatically be saved in the ICTEST.DAT file for future reference by the software.

TTL74.LIB

0000 0001 0002 0003 0004 0005 0006 0007 0008 0009 0010
0011 0012 0013 0014 0015 0016 0017 0018 0019 0020 0021
0022 0024 0025 0026 0027 0028 0030 0031 0032 0033 0034
0035 0037 0038 0039 0040 0042 0043 0044 0045 0046 0047
0048 0049 0050 0051 0053 0054 0055 0060 0061 0062 0063
0064 0065 0070 0071 0072 0073 0074 0075 0077 0078 0082
0083 0084 0085 0086 0089 0094 0095 0096 0107 0108 0109
0110 0111 0112 0013 0114 0116 0123 0125 0126 0128 0131
0132 0133 0134 0135 0136 0137 0138 0139 0140 0142 0145
0147 0148 0150 0151 0152 0153 0154 0155 0156 0157 0158
0159 0160 0161 0162 0163 0164 0165 0166 0168 0170 0173
0174 0175 0180 0182 0183 0189 0190 0191 0192 0193 0194
0195 0196 0197 0230 0231 0240 0241 0242 0243 0244 0245
0246 0247 0248 0249 0251 0253 0256 0257 0258 0259 0260
0265 0266 0273 0276 0278 0279 0280 0283 0290 0293 0295
0298 0299 0322 0323 0348 0351 0352 0353 0363 0364 0365
0366 0367 0368 0373 0374 0375 0376 0377 0378 0379 0386
0390 0393 0399 0465 0466 0467 0468 0490 0518 0519 0520
0521 0522 0533 0534 0540 0541 0563 0564 0573 0574 0575
0576 0577 0580 0590 0597 0620 0621 0622 0623 0638 0639
0640 0641 0642 0643 0644 0645 0646 0648 0668 0669 0670
0688 0689 0795 0796 0797 0804 0805 0841 0842 0870

CMOS40.LIB

0000 0001 0002 0006 0007 0008 0009 0010 0011 0012 0013
0014 0015 0016 0017 0018 0019 0020 0021 0022 0023 0025
0026 0027 0028 0029 0030 0032 0033 0035 0038 0040 0041
0042 0043 0044 0048 0049 0050 0051 0052 0053 0054 0055
0056 0060 0063 0066 0067 0068 0069 0070 0071 0072 0073
0075 0076 0077 0078 0081 0082 0085 0086 0093 0094 0095
0096 0097 0099 0101 0102 0103 0105 0106 0108 0109 0160
0161 0162 0163 0174 0175 0192 0193 0194

CMOS45.LIB

0001 0002 0003 0004 0006 0008 0010 0011 0012 0014 0015
0016 0017 0018 0019 0020 0029 0032 0038 0043 0053 0055
0056 0072 0082 0084 0085

DRAM

4164 41256 411000 41416 4416 4461 4464 41464
414256 44256 41400 44400(1Mx4) 441000(4Mx1)

SRAM

2147 2149 6116 6208(64Kx4) 6708(64Kx4) 6264
62256 2114/5114/9114 P4C187/L 628128(1M RAM)
584000(4M RAM) SQ8888 W24512(512K RAM) CY7C199

3. User Defined Test Vectors for Logic Device

A test vector is the definition of input and output logic states applicable to the various pins of an IC. Up to 1500 vectors may be specified for a single IC, thus allowing a sequence of logic events to be tested. There are 24 pins on the socket, i.e., socket pin 9 to pin 32 which can be defined as shown.

Definition of vector pin and Socket pin

```
Vector pin # 1  2  3  4  5  6  7  8  9  10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
Socket pin # 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
```

The syntax of a vector is as follows:

```
                                  1 1 1 1 1 1 1 1 2 2 2 2
Vector Pin No. 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4

V0001          X X X 0 1 N L H L 0 1 G 0 1 0 1 H L H L 0 X L E
                                                    <CR ><LF >
```

V0001 specifies vector line number

G	on pin 12 applies GND to IC.
E	on pin 24 applies 5 v to IC.
X	on pins 1, 2, 3, 22, means you do not care about input or expected output from IC.
N	on pin 6, means applied input and expected output are same as the last vector.
0	on pins 4, 10, 13, 15, 21 applies LOW to IC.
1	on pins 5, 11, 14, 16 applies HI to IC.
L	on pins 7, 9, 18, 20, 23 means expected output from IC is low on these pins.
H	on pins 8, 17, 19 means the expected output from IC is HI on these pins.

< CR > < LF > line termination codes are 0DH, 0AH (i.e., press ENTER)

NOTE

If, for example, a 14-pin IC is to be tested, then vector Pin 6 is electrically equivalent to Pin 1 of the IC and is also electrically equivalent to Pin 14 of the socket.

The Function test (single time) or Loop test is carried out according to the following sequence.

1. Apply G and then apply E to IC pins.
2. Apply 0, 1, N, X to pins
3. Read the value on each pin and compare it with the expected value defined in the vector.
4. If an error occurs, the error is displayed and the test is stopped.
5. If the test is successful, proceed to the next vector.

Vector restrictions:

Maximum number of vectors : 1500.
0, 1, L, H, X, N can be defined on any vector pin.
G can only be defined on vector pin 12.
E can only be defined on vector pins 19, 20, 22, 24.

Main Function Guide to the user defined vector test

Press " 4 " under the USER DEFINED TEST VECTOR menu to edit test vectors with the editor built into the software. The editing key functions will be shown on the screen. Otherword processors may be used provided they have an ASCII mode output, e.g., Wordstar in the N mode.

" 3 " saves the vector to the disk.

" 2 " loads the vector into the buffer.

USER function test and LOOP test.

"F" for vector function testing.
"L" for vector loop testing.
"D" will invoke the DISPLAY RESULT DURING TESTING function.

This function will display the test results on a single step basis or in a continuous mode if required, the results of each step can be output to the printer.

APPENDIX B

PLD VECTOR TEST

From the "Tester" item in the ACCESS menu enter item "3. PLD Vector" to activate test driver file (PALTEST.EXE). Through this function, the user can test PLD devices using JEDEC TEST VECTORS in the JEDEC fuse map file. A PLD vector test is a test where inputs are applied to the PLD and then the outputs of the device are tested for the correct function results based on the inputs. A PLD whose security fuses are blown can not be read but a vector test can be performed on the device to check its functionality. This function operates exactly the same as the user defined test vector function in the ICTEST function with a few different symbols. The available JEDEC TEST VECTOR symbols are listed as follows:

0	:	Apply LOW to PLD
1	:	Apply HI to PLD
H	:	HI output from PLD
L	:	LOW output from PLD
Z	:	Apply High-z to PLD
N	:	Apply power to PLD
X	:	Do not care
C	:	Apply positive clock
K	:	Apply negative clock

The maximum number of vectors in a test vector file is 1500.

NOTE

Although the PALTEST.EXE accepts the above vector symbols, we recommend that the user do not edit PLD vectors using any editors other than standard PLD assemblers or compilers such as ABEL, CUPL, OPAL, PLAN II, PDK-1, PALASM IV....

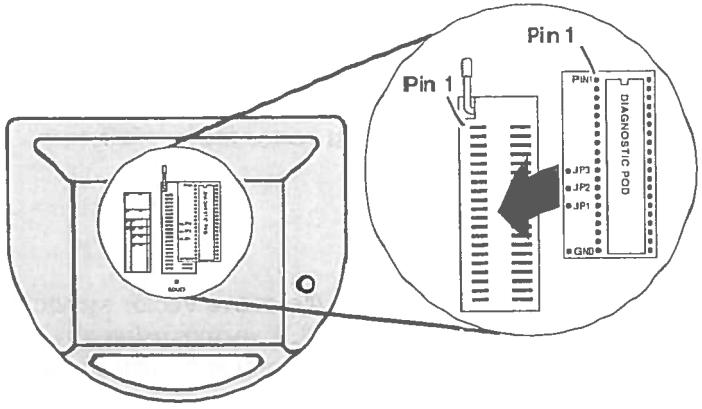
APPENDIX C

Programmer Self Test

The POD-DIAG40, a diagnostic test pod, is a tool for testing programming voltages of the programmer. The disgnostic test pod may be used if you encounter programming errors, or a device cannot be programmed. The testing package includes the diagnostic pod and a software diskette. This guide provides instructions for using the diagnostic test pod to test your programmer.

Inserting the Programmer Diagnostic Test Pod

To insert the diagnostic test pod, lift the lever 90 degrees (into the upright position) align pin 1 of the programmer and the diagnostic test pod, and insert the connectors of the diagnostic test pod into the programmer socket. When the diagnostic test pod is seated properly, gently push the socket lever forward 90 degrees to secure the connection with the diagnostic test pod.



Align Pin1 of the Programmer and Test POD

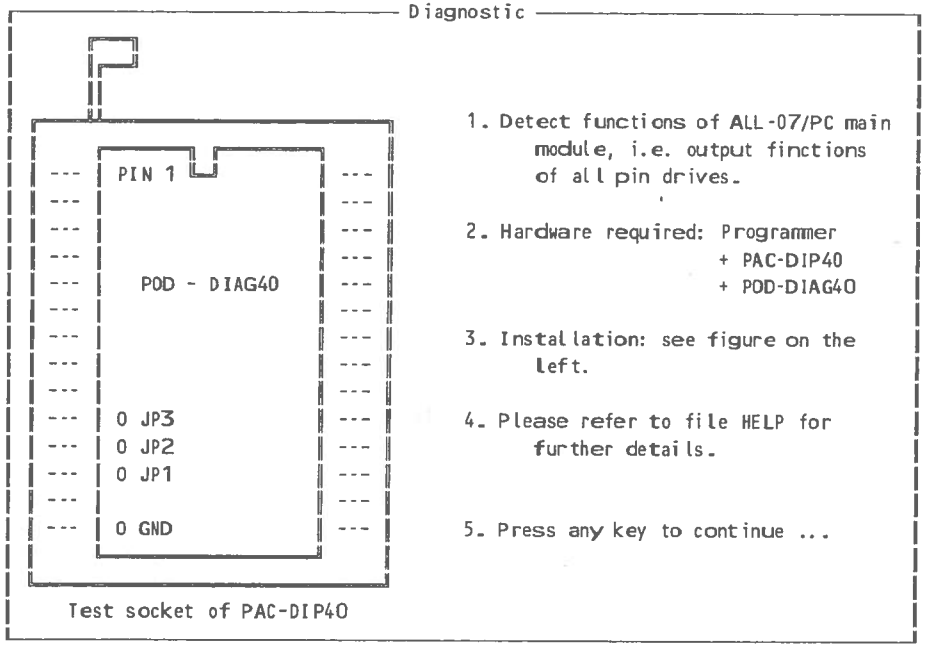
Launching the Diagnostic Software

1. Insert the POD-DIAG40 software diskette into the *A*:drive.
2. From the DOS prompt, launch the softwar by typing:

   A:\diag07 < enter >

Note: If you receive the message:
Error Identification on hardware!
Press <CR> to continue testing...
or press <Q> to Quit
Press <enter> to continue.
The message indicates a conflict with the base address.

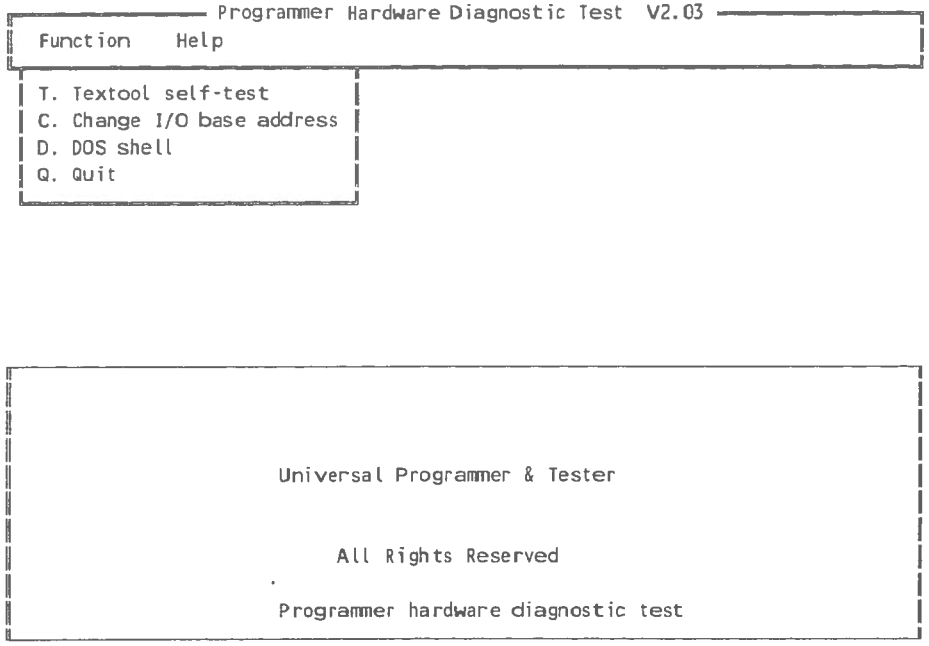
3. To change the base address, select Change I/O base address from the pull-down Function menu and select a new base address.
   This will launch the software and the following screen will appear:



Introduction Screen for the POD-DIAG40

Running the Diagnostic Software

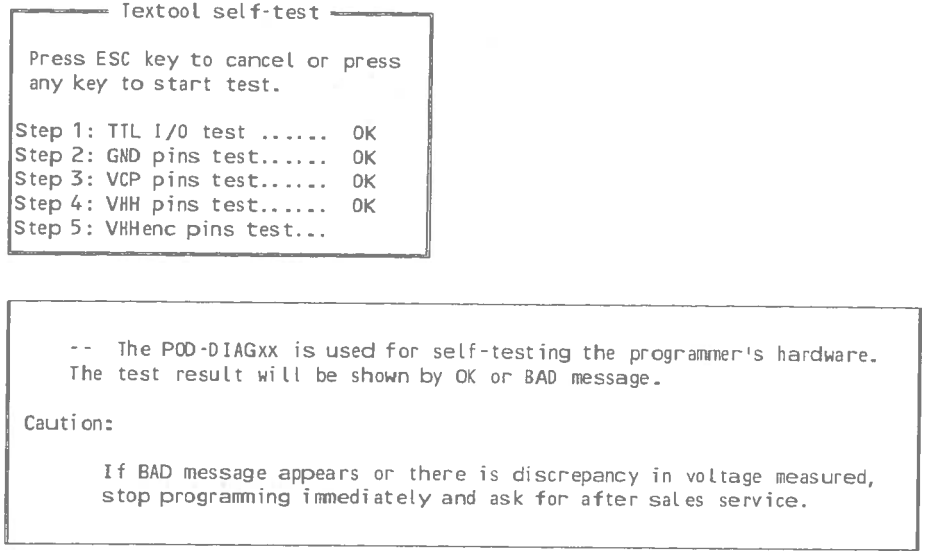
Once the software is launched, the Main Diagnostic Window appears. To run the programmer hardware diagnostic test, select the Textool self-test option from the Function menu.



Main Diagnostic Window

As the test program begins, you will see a reminder to insert the diagnostic test pod in the programmer. To cancel the test, press < Esc >. To continue the test procedure, press any other key.

While the test is running, the results will scroll through the upper left display window.

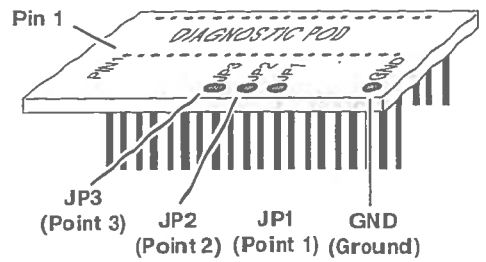


Viewing the Test Results in the Display Window

If a "Bad" message appears, repeat the test procedure to verify the results. If the results are the same after the second round of testing, the programmer needs to be sent in for calibration.

Using a Voltage Meter to Complete Your Test

Once the test is completed you may use a voltage meter to check the voltage of JP1, JP2, and JP3 (point1, point2, and point3) in relation to ground.



Locating the Voltage Test Points on the POD-DIAG40

To check the voltage, touch the ground probe of a voltage meter to the test point marked ground on the diagnostic test pod. Using the voltage probe, touch JP1 and check the readings against the voltage measurments displayed in the voltage measure window (refer to Figure of Viewing the Desired Voltage for Point 1, 2, and 3"). Complete the test by touching the voltage probe to JP2 and then to JP3 and comparing the readings to those in the Voltage Measure window.

```
Textool self-test
Step 1: TTL I/O test ......  OK
Step 2: GND pins test......  OK
Step 3: VCP pins test......  OK
Step 4: VHH pins test......  OK
Step 5: VHHenc pins test...  OK
Step 6: VOP pins test......  OK
Step 7: VOPX pins test.....  OK
Step 8: OSC pins test......  OK
```

```
Voltage Measure

   Use multimeter with accuracy
   +/- 0.1%+1 and 4.5 figures.

 Point 1: JP1 - GND   8V ± 0.1V
 Point 2: JP2 - GND  12V ± 0.1V
 Point 3: JP3 - GND  21V ± 0.1V

Complete checking by pressing any key.
```

```
   --  The POD-DIAGxx is used for self-testing the programmer's hardware.
   The test result will be shown by OK or BAD message.

Caution:

      If BAD message appears or there is discrepancy in voltage measured,
      stop programming immediately and ask for after sales service.
```


Available On-line Help

If you have any questions while you are running the software for the POD-DIAG40, on-line help is available

```
Programmer Hardware Diagnostic Test  V2.03
Function    Help
```

```
I. Introduction
G. Getting Started
D. Diagnostic
C. I/O port Configuration
S. DOS Shell and Quit
```

```
                    Programmer

              Universal Programmer & Tester



              Programmer hardware diagnostic test
```

APPENDIX D

TROUBLESHOOTING

PROBLEM 1

When I turn my computer on, I get no beeps, the fan does not spin, nothing happens!

RESOLUTION

1-1 The power cord may be disconnected from the computer or the wall. Check the power cable.

1-2 You may have a chip improperly inserted in the ZIF socket. Make sure your chip is correctly installed and the socket lever is pressed down.

1-3 Your power supply may not have sufficient power to drive both your system and the interface card.

PROBLEM 2

When I try to run a driver file, I get communication error messages!

RESOLUTION

2-1 You may not have the I/O port set correctly for your programmer. Double check the I/O port assignment.

2-2 You may have a chip improperly inserted in the ZIF socket. Make sure your chip is correctly installed and the socket lever is pressed down.

2-3 There may not be bad connection between the interface card and the programmer. Double check the cable connection.

2-4 Your system may be running too fast. Try slowing your system down as much as possible. For example, set clock of AT Bus or ISA Bus at 7.159 MHz in the PC BIOS setup.

PROBLEM 3

When I install the interface card, some of my other peripherals start behaving strangely!

RESOLUTION

3-1 You are probably experiencing an I/O port conflict. Double check the I/O port assignments on all your peripherals, including the interface card.

PROBLEM 4

Two or more programming failures in a row.

RESOLUTION

4-1 Ask your dealer if any Spec. revision has already been updated to the driver file.

4-2 Make sure you are running the programmer software directly from DOS. If you run from a DOS box in Windows 3.x or Windows 95 you may have problems.

10 Things to do Before Calling Your Dealer

1. Reboot the computer and try again.

2. If you change switches or jumpers, write down the original settings.

3. Repeat all the steps, following the instructions in this manual.

4. Make sure all cards and cables are firmly attached.

5. Remove any memory resident programs from memory.

6. See if your problem is listed in the Trouble-Shooting section.

7. Try it on another system.

8. Compare system requirements with your configuration.

9. Ask your in-house guru (every office has one).

10. Ask whoever installed the product.

GENERAL TROUBLESHOOTING CHECKLIST

If your problem is not described above, check the following :

1. Is the interface card fully seated in its slot?
2. Are all cable connections securely fastened?
3. Does the interface card jumper have the same I/O address setting as in the driver file?
4. Does any other card on the bus have the same I/O address as the interaface card?