USER'S MANUAL
EXPRO-60/80

PC-BASED UNIVERSAL PROGRAMMER

SUNSHINE

USER'S MANUAL

EXPRO－60/80

PC－BASED UNIVERSAL

PROGRAMMER and TESTER

Version : V4.2
Manual released AUGUST 1992

SUNSHINE

Programmer Module :**EXPRO－60/80**
System Adapter Card :**SAC－101A**

CONTENTS

5. USING THE UTILITY FILES

APPENDICES

1. INTRODUCTION

1.1 Manual Contents

This manual describes the methods of installing and operating the UNIVERSAL Programmer with an IBM PC running MS - DOS or PC - DOS. This manual also contains information about the UNIVERSAL Progrommer's usage and detailed functions.

The user is assumed to be an experienced user who is familiar with general software installation on PC.

1.2 Product Configuration

Before using this product, please check carefully that your package includes:

* PC System Adapter Card.
* 1 - M. cable with two 25 - pin, D - type connectors on cable's end.
* Programmer module.
* User's manual.
* 1 diskette including the following files:

README1.DOC	:Software & hardware revision message.
EXPRO.EXE	:Auto device driver file.
EPP512.EXE	: Device driver file for EPPROM under 512K bits.
EEP1.EXE	:Device driver file for EEPROM.
SEEP1.EXE	: Device driver file for 8 pin serial EEPROM and special serial PROM.
EPP1024.EXE	:Device driver file for EPROM over 1 M bits.
BPPGM.EXE	:Device driver file for BIPOLAR PROM.
README2.DOC	:Software & hardware revision message.
PALPx.EXE	:Device driver file for PAL device.

x:from 1 to 5.

README3.DOC	:Software & hardware revision message.
EPLD1.EXE	: Device driver file for ALTERA, TI, INTEL EPLD.
GAL1.EXE	:Device driver file for LATTICE, NS GAL.
GAL3.EXE	:Device driver file for LATTICE, NS GAL6001.
GAL2.EXE	:Device driver file for AMD PALCE Series.
PEEL1.EXE	:Device driver file for ICT PEEL.
README.DOC	:Software & hardware revision message.
PALTEST.EXE	:PLD VECTOR TESTING program.
ICTEST.EXE	:IC TESTER program.
TTL74.LIB	:TTL 74 series library.
CMOS40.LIB	:CMOS 40 series library.
CMOS45.LIB	:CMOS 45 series library.
7406.VEC	:Sample vector for TTL 7406.
4040.VEC	:Sample vector for CMOS 4040.
UTIL <DIR>	:Subdirectroy of UTILITY files includes.
DUMP.EXE	:Dump file to Console in BINARY format.
HEXBIN.EXE	:HEX in BINARY file converter.
HEXBIN2.EXE	:EXTENDED HEX to BINARY converter.
SPLIT2.EXE	:2 - way file splitter.
SPILT4.EXE	:4 - way file splitter.
SHUFF2.EXE	:2 - way file shuffler.
SHUFF4.EXE	:4 - way file ahuffler.
EDBIN.EXE	:Edit a binary file.
SEP2MB.EXE	:Seperate a BINARY File large than 2Mbits
README5.DOC	:Software & hardware revision message.
PGM48.EXE	:Device driver file for 8748 series MPU.
PGM51.EXE	:Device driver file for 8751 series MPU.
PGMZ8.EXE	:Device driver file for Z8 series MPU.

MPU1.EXE	:Device driver file for HITACHI 63701V0, TI 7742, 77C82.
FPLP1.EXE	:Device driver file for SIGNETICS FPL device.
FPLP2.EXE	:Device driver file for SIGNETICS FPL device.
EXPRO.INF	:Device information list.
PRODUCTS.LST	:Introduction of all SUNSHINE product.

2. INSTALLATION

This chapter describes the method of installing an UNIVERSAL programmer on an IBM PC/XT/AT/386 computer or compatible running MS－DOS or PC－DOS.

It is assumed that the installer is familiar with the installation of PC add－on cards and software in running MS－DOS.

2.1 Host System Requirements

* IBM PC/XT/AT/386 or compatible PC.
* Max. system clock speed is up to 25 MHz, Turbo mode.
* Min. 640K bytes memory.
* Min. 1 floppy disk drive. A hard disk is preferable.
* Operation system ：MS－DOS or PC－DOS , version 2.0 or later.

2.2 Software Installation Procedures

2.2.1 PC system has one hard disk drive and at least one floppydisk drive.

The hard disk installation procedure is very simple. Please follow the listed steps to copy all the files on the supplied diskettes to a subdirectory on the hard disk.

Steps as follows：

	Description
C:》md EXPRO	Generate a programmer subdirectory(EXPRO)
C:》cd EXPRO	Change directory to EXPRO
C:EXPRO》copy A:*.* C:	Copy all files from A: to the current directory in C:

To copy the utility file under subdirectory UTIL of diskette, please follow the steps listed below.

	Description
C:EXPRO》md UTIL	Generate a subdirectory UTIL under EXPRO
C:EXPRO》cd UTIL	Change directory to UTIL
C:EXPRO\UTIL\》copy A:UTIL C:	copy the UTIL flie from A: to C:

Whenever you have a new updated diskette in hand, by applying the above mentioned methods, you can easily perform your own updating.

Then, proceed with the Hardware Installation procedure.

2.2.2 PC System with Floppy Disk drive only.

Please proceed with step 2.3 Hardware Installation Procedure, as no software installation is needed under this configuration.

2.3 Hardware Installation Procedures

Before starting the installation procedure, it is necessary for the user to duplicate the original software on to a working disk and use it for programming. Do not use the original software diskette ! If you attempt to use the original diskette, your monitor will display a "disk write error" message whenever you terminate the device driver file.

To install the UNIVERSAL Programmer and the software supplied, follow these steps.

Step 1:

Switch off your computer system, and open the computer cover carefully.

Step 2:

Check the DIP switch of the PC system adapter card.

For I/O address selection. **2E0H(default)**

SW1: all off.

Sw2: position 7 on, others off.

For I/O wait state selection.

SW3 position 1 on, 8 waits.
position 2 on, 4 waits(default).
position 3 on, 2 waits.
position 4 on, 1 wait.

Step 3:

Insert the system adapter card gently into the PC slot, and fasten it to the PC frame with the slot cover screw.

Step 4:

Connect the programmer module to the system adapter card using the attached cable. The male cable end must be inserted into the system adapter card, while the female end must be connected to the programmer module.

CAUTION

Do not connect the programmer module to the system adapter card when the computer is turned on. Such an installation can put the module in an "unknown" state, resulting in damage to the DEVICE and/or the module.

Step 5:

Turn on the computer and check LEDs on programmer module.

ON LED must be **ON**.
BUSY LED must be **OFF**.
Other LEDs are in random state.

If the LEDs are not in the correct state, please turn off the PC and check all connections between the system adapter card and PC slot and cable connections between the system adapter card and programmer module. Then turn on the computer and check the LEDs on programmer module again.

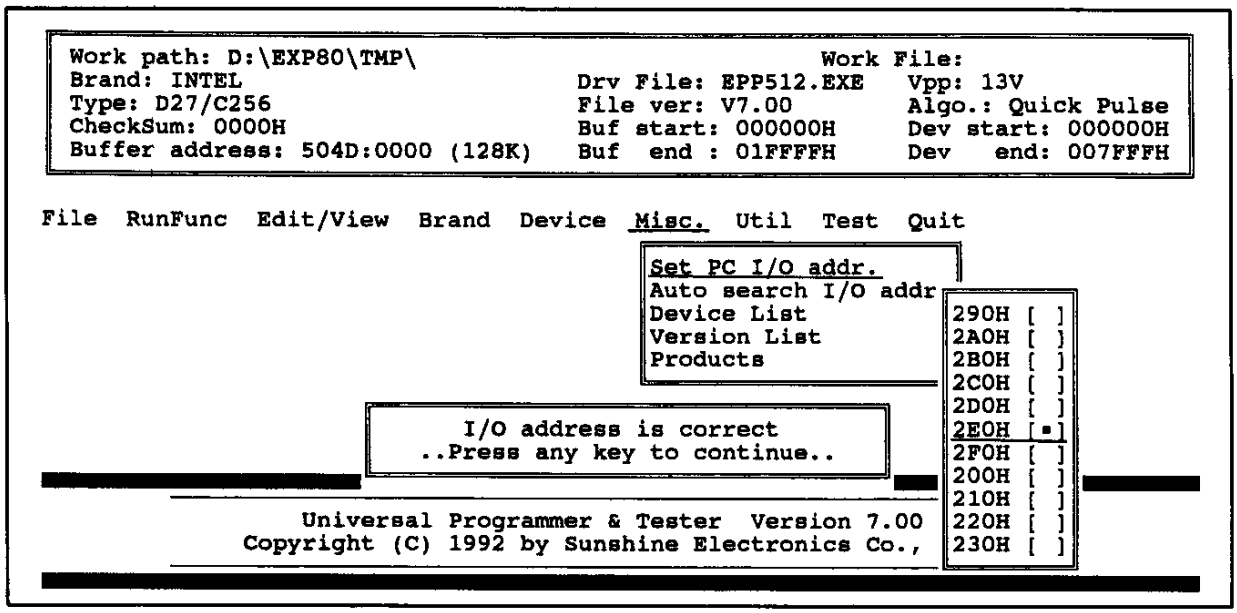
Step 6:

Boot the DOS and then perform "**Set PC I/O Base Address**" function according to the following ways:

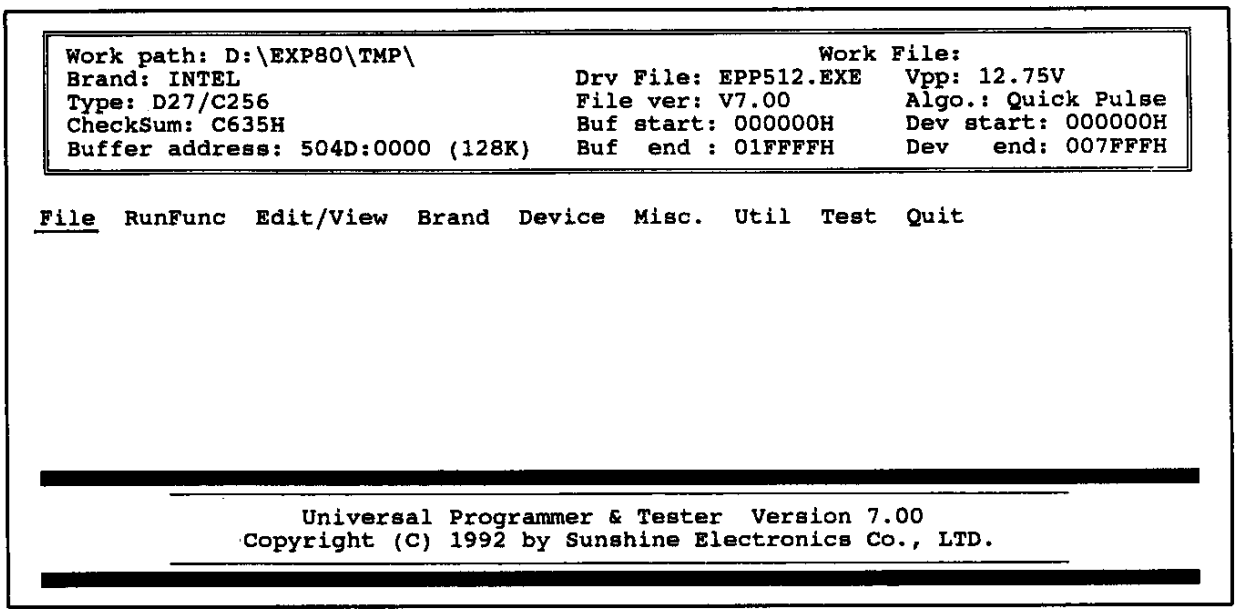
Step 6.1: Take subdirectory "**EXPRO**" as an example , type the command "**EXPRO**" under its subdirectroy.

C:EXPRO》EXPRO

After executing the "EXPRO" command, the following **MAINMENU** will appear:



Type '**M**' or move the highlighted bar to "**Misc.**" to select" **Set PC I/O addr.**", and the following sum – menu will appear:



The current I/O base address registered in the software will appear in the center of the screen, and the software will automatically check with the identification circuit in the programmer module. The software will display **"I/O address is correct"** when the result is a correct match, otherwise, **"Communication error"** will appear.

Step 6.2

"Communication error" appears when the hardware I/O address of the system adapter card is mismatched with the software default I/O address, or when the hardware I/O address of the system adapter card conflicts with other add－on cards inserted into the PC. Press **'Q' or <Esc>** to quit to DOS prompt, and then turn off the computer and re－check all the connections between the system adapter card and the PC, and all cable connections between the system adapter card and the programmer module.

After re－checking all connections, repeat the installation

procedure from step1.

NOTE

Most problems are due to poor contacts between the system adapter card and the PC slot, or cable connections between the system adapter card and programmer module.

Step 6.3

If the software I/O address is identical to that of any other add－on card, the user may directly select the I/O address on the screen, and change the hardware I/O address as well to match both software and hardware.

Please refer to Section **4.18 Change I/O Base Address function** to setup a new I/O address.

3. USING THE PROGRAMMER

In this chapter we will step by step help the user familiarize with his new UNIVERSAL Programmer tools. It is assumed that the Installation Procedures described in the previous chapter has already been performed.

3.1 Definition of Symbols

The most commonly used symbols are here defined and explained to enhance usage and for easy reference.

3. 1. 1 EPROM, EEPROM, BPROM and MPU Device symbols

*** BIT, NIBBLE, BYTE, WORD, ADDRESS and BUFFER BASE ADDRESS**

BIT :An element of Binary data.
NIBBLE :A 4－bit Binary data Value from 0h to FH.
BYTE :A 8－bit Binary data Value from 0h to FFH.
WORD :A 16－bit Binary data Value from 0h to FFFFH.
ADDRESS Location of data on buffer. Value from 0H to FFFFFH.
BUFFER BASE ADDRESS:
The actual physical address of the buffer.
Value from 0000:0000 to F000:FFFF.

*** Memory Buffer**

The buffer is a block area of PC memory allocated by the device driver file through DOS. This buffer is used by the device driver flie as an intermediate storage.

The device driver file can read the DEVICE contents to the buffer and save it on a disk file or perform the reverse operation, that is to load a disk file to the buffer and program it to the DEVICE. You can easily manipulate the buffer contents at will.

The modified buffer can always be saved to a disk file for future reference.

The minimum size allocated to the buffer is 64K bytes, and the maximum size is the maximum available memory in the PC.

Since the size of the memory buffer is allocated by 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 memory buffer.

* Buffer Start and Buffer End Address

The buffer start and end address are offset addresses specified from the base address of the MEMORY BUFFER. This is the specified portion where information can be programmed to the DEVICE and the DEVICE contents can be read onto the buffer.

* Device Start and Device End Addresses

The start and end addresses are offset addresses specified on the DEVICE contents.

* Check Sum

This is the sum of all data contents between buffer start and end addresses. This value 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.
MCU is 8.

* EVEN and ODD address mapping

The address sequence of data contents can be assigned to be CONTIGUOUS, EVEN or ODD whenever you want to READ, PROGRAM or VERIFY the EPROM.
For example, in a program function sub－menu, the screen displays a query,

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 reverse direction.

* I/O Address

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

This default I/O base address of the system adapter card is **2E0** and occupies 4 contiguous spaces(**2E0** to **2E3**).

* Counter

This is the programming address counter. During the DEVICE programming, the counter value will be accumulatingly displayed on screen.

* 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 Special Device PAL, FPL, GAL, PEEL and EPLD Symbols

* Programmable Logic Device (PLD)

Generally speaking, **PLD** is a device that can be programmed to perform many different logic operations.

PLDs are grouped into the following three categories:

PLD :A one time PLD such as a **PAL** deivce.
EPLD :A UV erasable PLD such as a **EPLD** device.
EEPLD : An electrical erasable PLD such as a **GAL**, or **PEEL** device.

* 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 fuses information (BLOWN/INTACT) and the FUNCTION TEST VECTOR of the PLD. Most PAL assemblers or compilers, such as the PALASM, PLAN, OPAL, ABEL, SNAP, AMAZE and PDK－1, will produce a JEDEC fuse map file.

* **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 as COMBINATORIAL/REGISTERED, OUTPUT FEEDBACK/OUTPUT ENABLE.

* **Security Fuses**

Compared to PROM device, there are extra fuses inside the PLD. The arrangement of array fuses and configuration fuses will normally not be copied if the security fuses are blown.

* **Fuse BLOWN and INTACT**

Originally in the JEDEC fuse map definition, the new PLDs (Blank devices) are all in a fuse **INTACT** state (i.e., **0 state**). The user can only blow an INTACT fuse into a BLOWN state(i.e., **1 state**).

Due to the rapid improvement of PLD technology, some new PLDs are in a fuse BLOWN state, and can only be programmed into an INTACT state. There is no way to return to the initial state unless the PLD is erasable.

3.2 Viewing the Main Menu

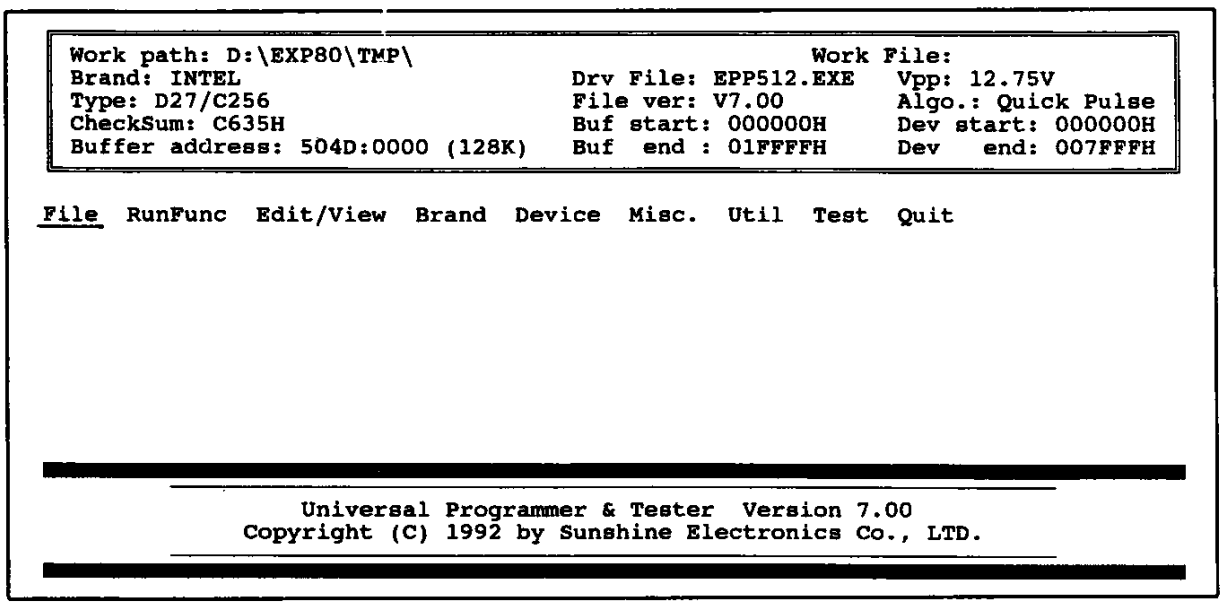
First, study the **AUTO device driver** file's main menu displayed on the screen to familiarize yourself with the

menu – driven features.

3.2.1 Status Field

On the top of the following screen is a group of text. Inside the window is the current DEVICE's **BRAND, TYPE, memory BUFFER ADDRESS, DEVICE ADDRESS, driver file's VERSION , CHECKSUM , WORK filename, VPP and Programming ALGORITHM**. This is the status field.

Whenever you are going to use this programmer, please make sure that the status in this field meets your requirements. Otherwise the DEVICE will be **destroyed** or
programmed to an unknown state.



3.2.2 Function field

Directly below the status field is the function field which includes

File, Runfunc, Edit/View, Brand, Device, Misc., Util, Test, Quit.

3.2.3 Logo, Hardware Model and Software Version

At the bottom of the screen, you will see the **LOGO, HARDWARE MODEL** and **AUTO DRIVER FILE' S VERSION**. These messages are for reference only.

By now you should be familiar with the main menu screen. You may learn more about these features by reading the rest of this section. Functions will be discussed in detail.

3.3 Getting Started

This section presents a simple example to help you familiarize with the programmer and some commonly used functions. You can refer to Chapter 4 for detailed description of each function. If you did not get the desired results, make sure that all the cables are connected firmly. Always double check cable connections.

The follwing illustration will demonstrate how to start and exit the main menu and operate functions such as File, Runfunc, Edit/View, Brand, Device, Misc., Util and Test.

In the illustration, we shall use an **INTEL 27C256** EPROM. You may also use other series PLD or EPROM devices to practice this exercise.

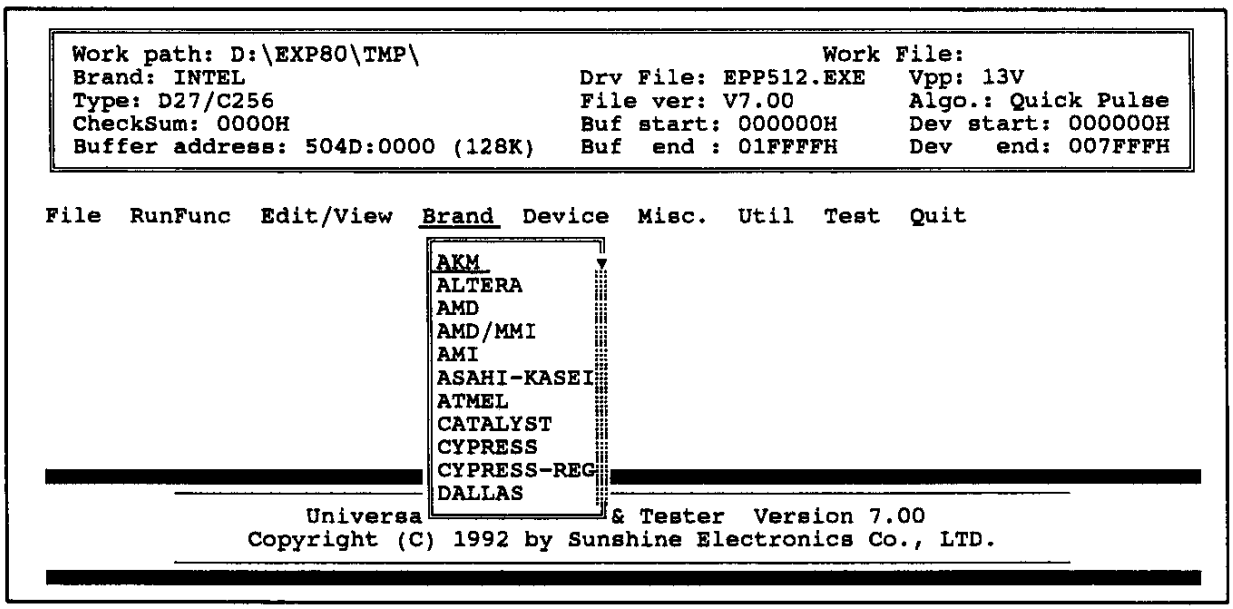
Now, if you have at least one **INTEL 27C256** EPROM in hand, you can begin by taking the following steps.

3. 3. 1 – 1 Execute the EXPRO program, type the "**EXPRO**" command under the subdirectory **EXPRO**.

C:》EXPRO

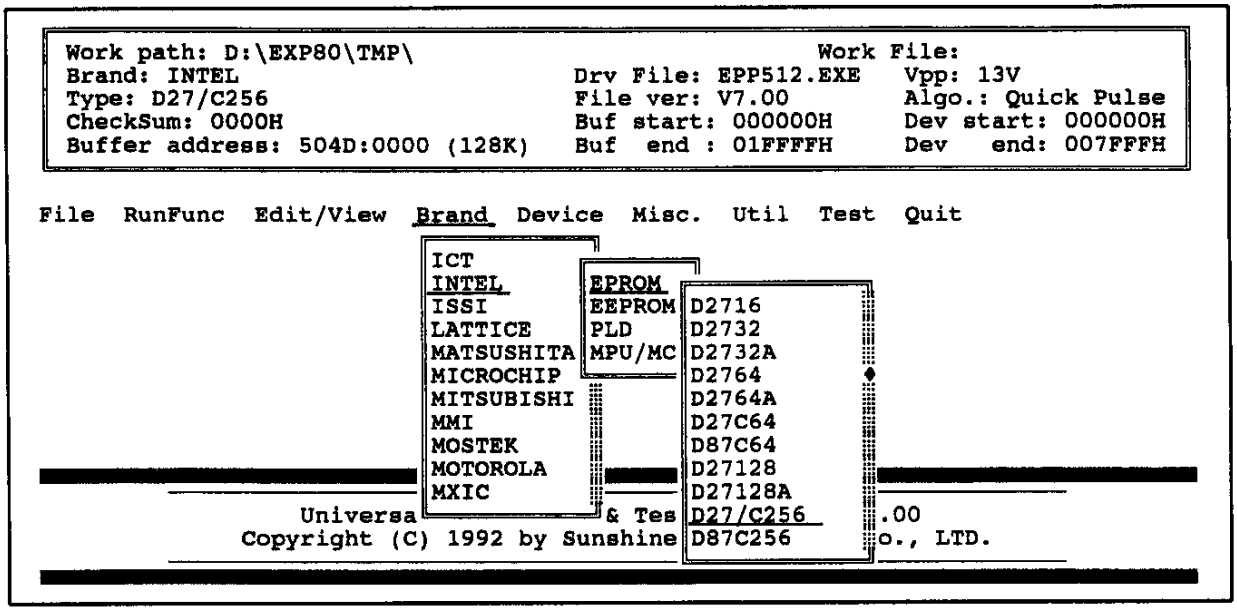
After entering the "**EXPRO**" command, the following MAIN MENU appears.

Type '**B**' or move the highlighted bar to '**Brand**' to select **MFR**.
The following **MFR** list will appear on the screen, as follows:



3.3.1－2 Select EPROM MFR and TYPE.

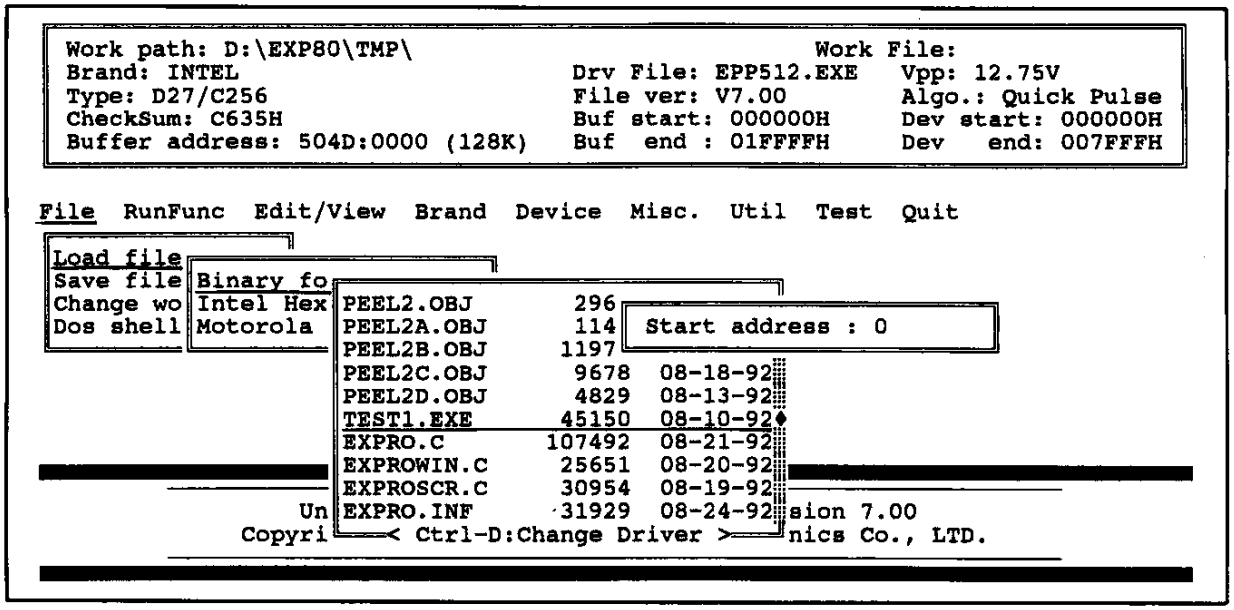
Type '**I**' or move the highlighted bar to select "**INTEL**", then type '**R**' to select "**EPROM**" . The screen will appear the following:



Move the highlighted bar to select "**27C256**". After selecting the correct type, the EXPRO program will automatically return to the "**Runfunc**" and modify **STATUS FIELD** as you have selected.

3.3.3 Load Disk File into Buffer.

After you have taken the above mentioned steps, type '**F**' to select "**Load file**" function to transfer the file in **BINARY** or **HEX** format to the memory buffer, and the following file loading sub－menu will appear：



sub－menu of file loading

Please follow the listed steps to execute the window selection:

. Use the <**UP**> or <**DOWN**> or <**PgUp**> or <**PgDn**> keys to move the highlighted bar to the desired file or subdirectory.
. Press the < **Enter** > key to select the desired file or sub directory.
. If the subdirectory name is selected, the directory window will display all the files of that subdirectory for next selection.

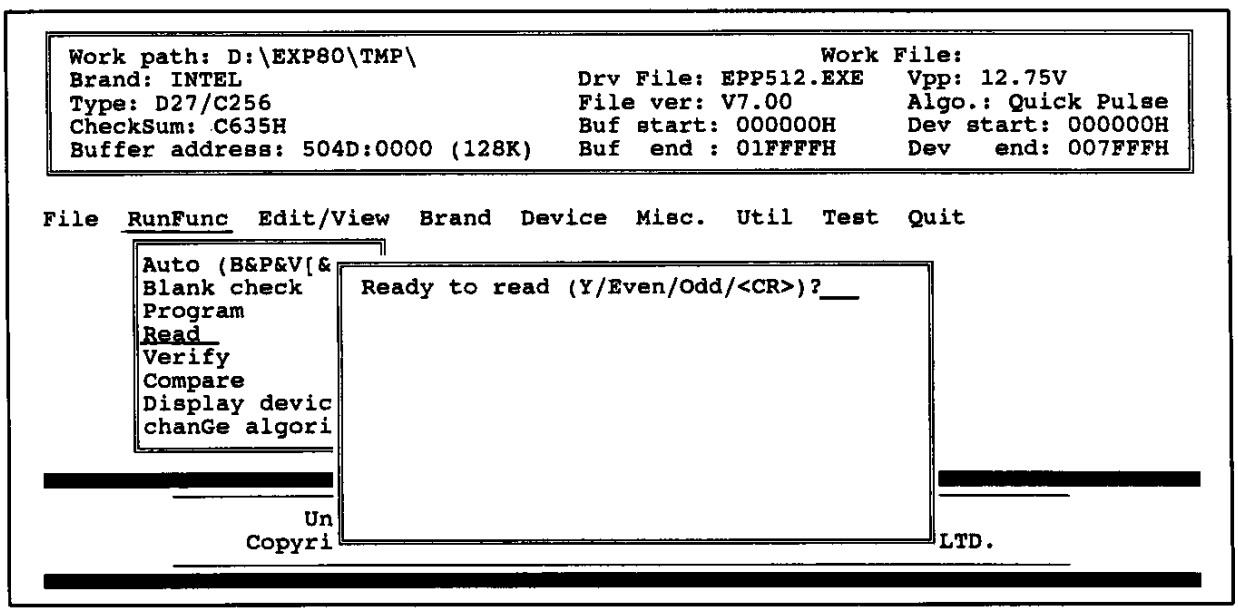
After entering the file name, the sub－menu will require you to enter the buffer starting address that you want to load from. You must now specify the memory buffer location (address) to which the disk file is to be sent. Type 0000 if you wish the disk file to be transferred to the starting position of the memeory buffer. After entering the location, the software will begin to load the disk file to the internal memory buffer and display.

LOADING NOW...
OK!

The disk file is now in the memory buffer. Press <**Esc**> or <**CR**> to return to the "File".

3.3.4 Read Contents from Master EPROM.

If the EPROM data is in a Master EPROM instead of a disk file, you have to transfer it by typing 'R' to select "RunFunc". After typing 'R' or move the highlighted bar to select "Read" function , the following sub－menu will appear:



sub－menu of READ

Insert the **Master EPROM** into the socket. With the pull lever in upright position, the lower left pin should match the

EPROM GND pin. Then press **'Y'**, the data of the Master EPROM will be transferred to the internal memory buffer and diaplay:

READING NOW...
OK!

Press <**Esc**> or <**CR**> to return to the "Runfunc".

3.3.5 Insert the Blank EPROM into Textool.

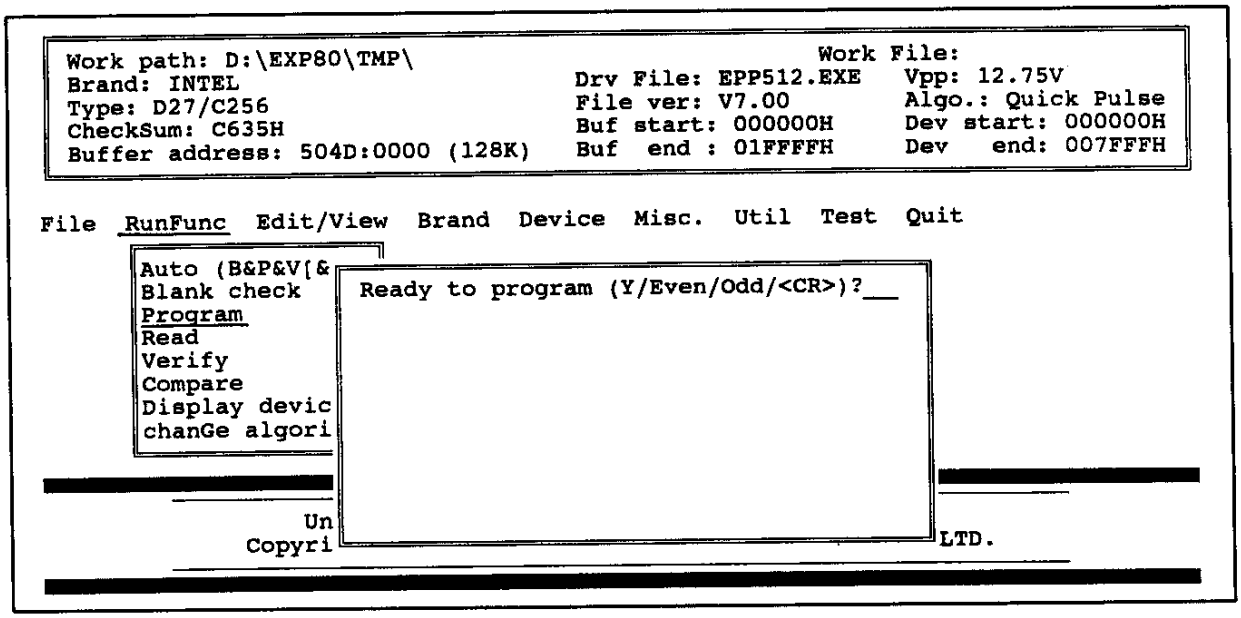
After transferring the data from the disk file of the Master EPROM, take out the Master EPROM and insert the blank EPROM into textool with the pull lever in upright position. The lower left pin should match the EPROM GND pin.

CAUTION

The notch - end of the EPROM must not be inserted into the end of the socket, if not, the EPROM will be **destroyed** or programmed to an unknown state.

3.3.6 Program Buffer Contents to EPROM.

After loading the disk file or reading the Master EPROM data into the memory buffer, and inserting the **blank EPROM**, you can now program the EPROM by typing **'R'** to select "**RunFunc**". After typing **'P'** or move the highlighted bar to select "**Program**" function , the following sub - menu will be displayed:



sub – menu of Program

Then type **'Y'**, and the programmer will attempt to program the buffer contents onto the **blank EPROM**. At the end of the programming process, the programmer will compare EPROM contents with the memory buffer. Any discrepancies between the buffer and the EPROM will be displayed.

This completes the programming process. To program other EPROMs, wait for the **BUSY LED** to shut **off**, then replace the EPROM and type **'Y'** again.

If you want to exit the programming process, press **<Esc>** or **<CR>** to return to the "Runfunc".

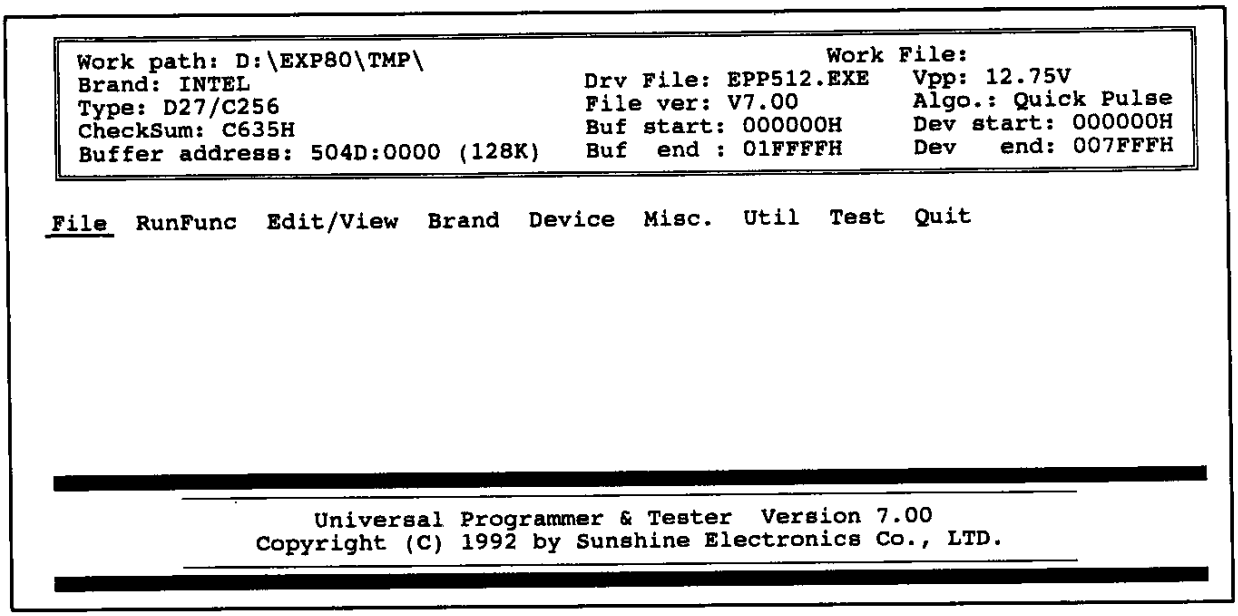
We hope that with the aid of this manual, you can become a professional user of this programmer module without difficulty.

The advanced user may want to modify the contents in the

memory buffer, save the buffer contents in the disk fill, or change the I/O address to another address...etc. We will describe each of the functions in detail in Chapter 4's FUNCTION REFERENCE GUIDE.

4. FUNCTION REFERENCE GUIDE

The following reference guide describes each of the functions found in the main menu. The functions will be described in the order given in the menu. To enter into the desired function, press the **first letter** of that function.

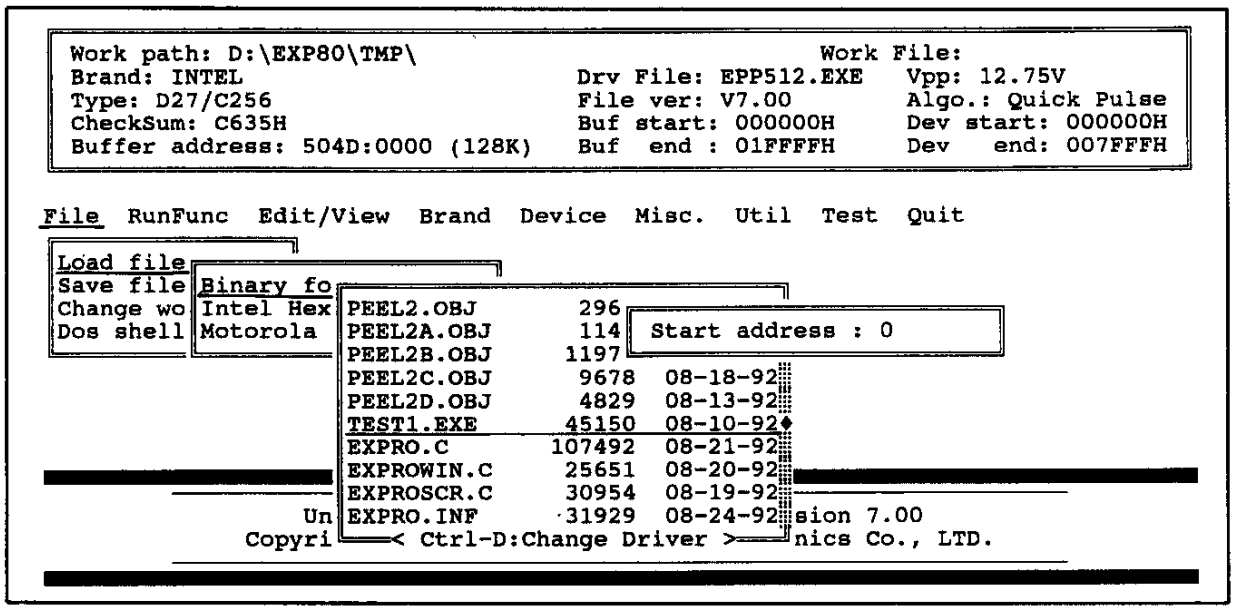


Screen of main menu

4.1 Load BINARY/HEX File TO buffer

Type '**F**' to select "**File**" followed by '**L**' to select "**Load file**"; or move highlighted bar to select "**File**" and "**Load file**".

A dialog window will be displayed on the screen as follows:



Load file

. Use the <**UP**> or <**DOWN**> or <**PgUp**> or <**PgDn**> keys to move the highlighted bar to the desired file or subdirectory.

. Press the < **Enter** > key to select the desired file or subdirectory.

. If the subdirectory name is selected, the directory window will show all the files of that sub－directory for the next selection.

. Press '**Ctrl+D**' to change disk driver.

After entering the file name, the sub－menu will require you to enter the buffer starting address that you want to load from. You must now specify the memory buffer location (address) to which the disk file is to be sent. Type 0000 if you wish the disk file to be transferred to the beginning of the memeory buffer. After entering the location, the software will begin to load the disk file to the internal memory buffer and displays.

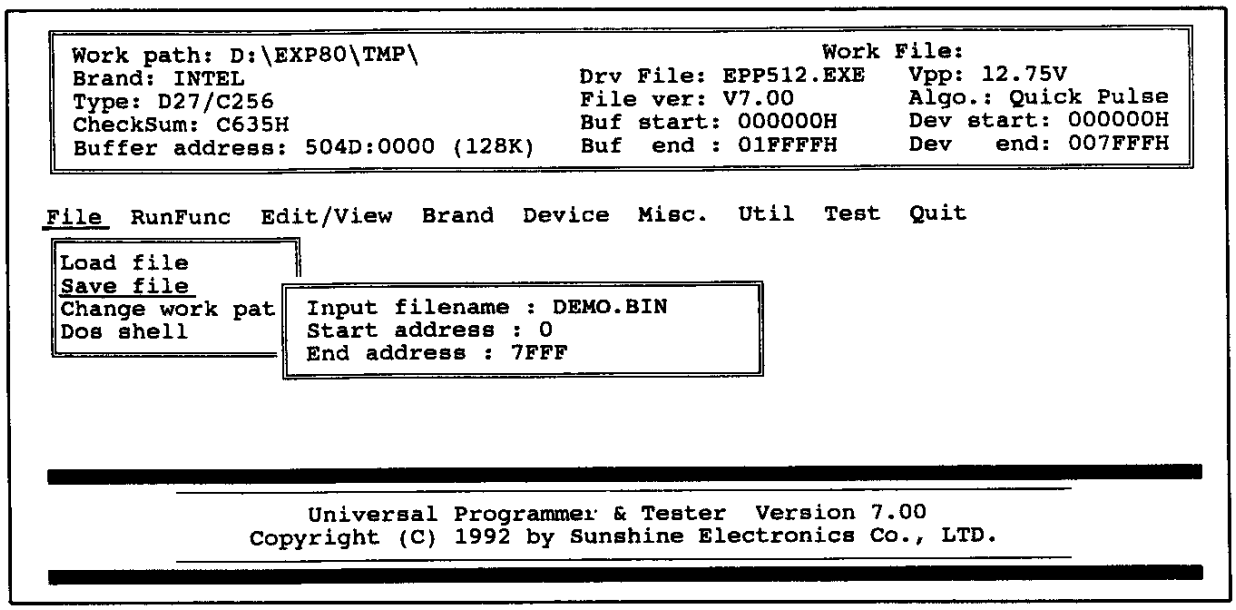
LOADING NOW...
OK!

Press <**Esc**> to return to the "File".

4.2 Save Buffer to Disk

Type '**F**' to select "**File**" followed by '**S**' to select "**Save file**"; or move highlighted bar to select "**File**" and "**Save file**".

A dialog window will be displayed on the screen as follows:



Save file

When prompted, key in the complete **filename** to be saved, then press <CR>.

Next, key in the starting address for the portion of buffer to be saved, and press <CR>. Then, key in the ending address for the

portion of buffer to be saved, and press <CR>.

The buffer has been saved when the OK! message appears. Press <**Esc**> to return to the "File".

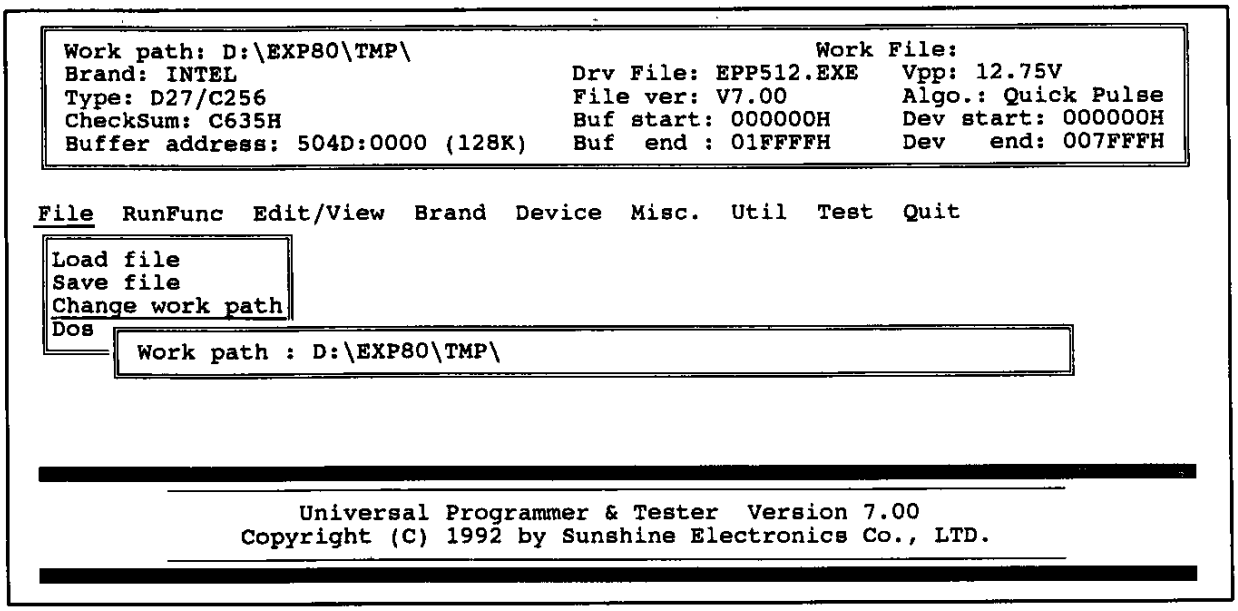
NOTE

Files saved under this function are exact Binary file with no conversion.

4.3 Change work path

Type '**F**' to select "**File**" followed by '**C**' to select "**Change work path**"; or move highlighted bar to select "**File**" and "**Change work path**".

A dialog window will be displayed on the screen as follows:

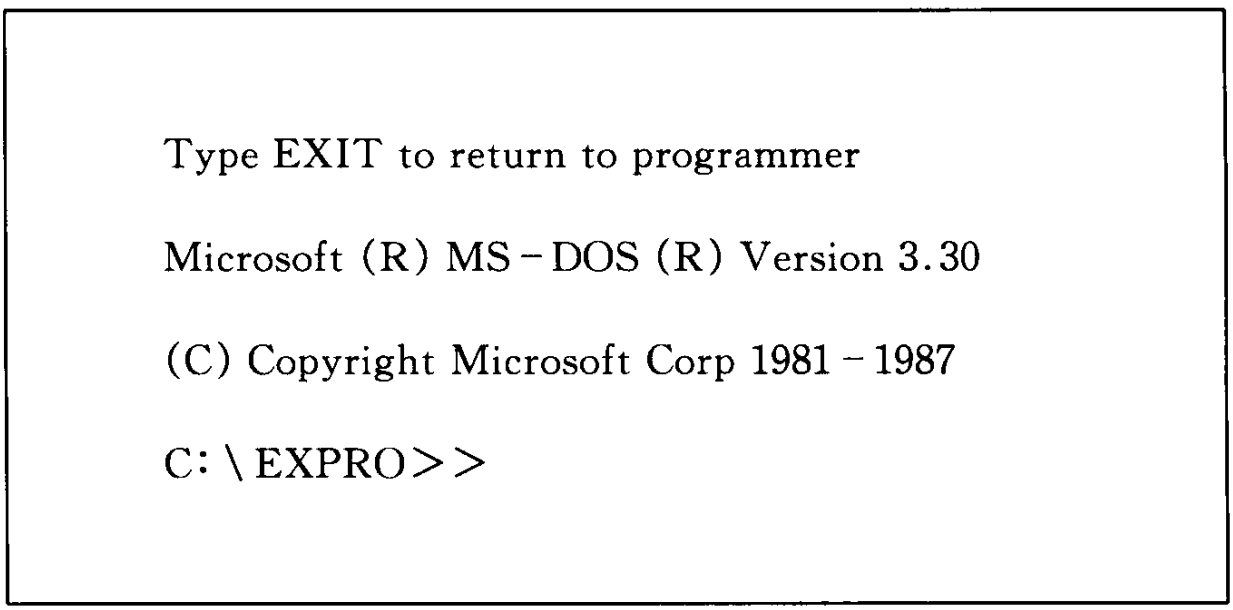


Change work path

When prompted, key in the complete name of directory, then press <CR>.
You can change to any directory to load and save file.

4.4 DOS Shell

Type '**F**' to select "**File**" followed by '**D**' to select "**Dos shell**"; or move highlighted bar to select "**File**" and "**Dos shell**".
After entering into this function, the software will search for the **COMMAND. COM** in the DOS boot disk drive. If it exists, the software will execute this file, pass control to it and display a DOS command prompt as follows:



DOS Shell

By now, the software is controlled by DOS and waiting for your command. The command format is the same as the DOS

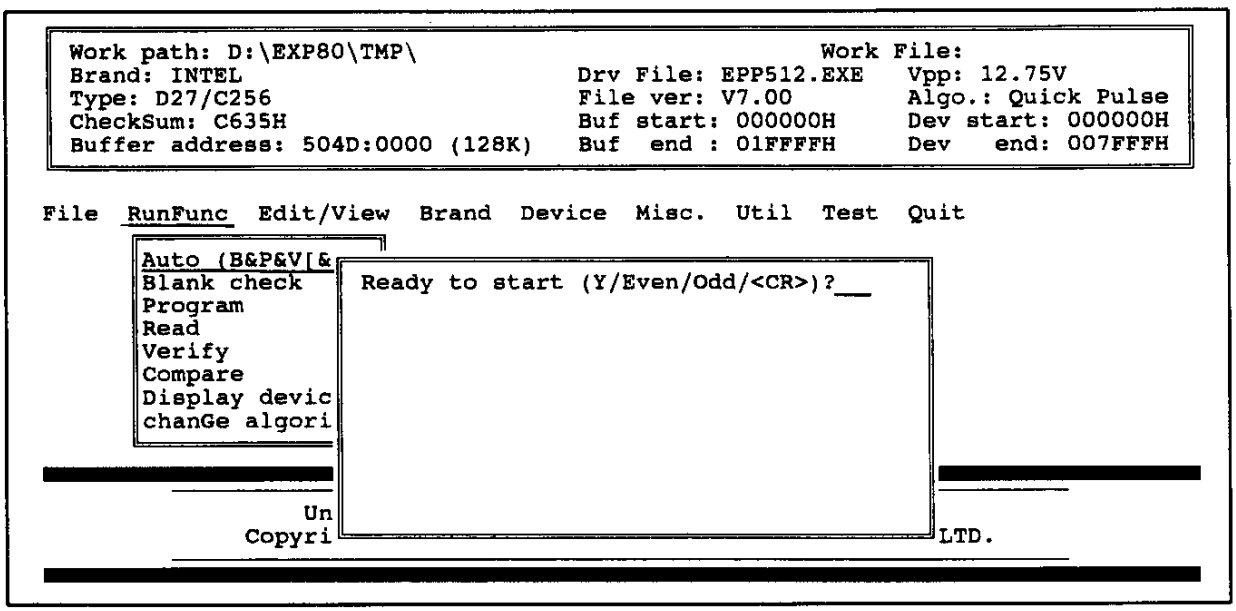
command.
Under DOS command prompt, enter both text **'EXIT'** and <CR> to return to the "File".

NOTE

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

4.5 Auto (B & P & V [&S])

Type **'R'** to select **"Runfunc"** followed by **'A'** to select **"Auto (B & P & V [&S])"**; or move highlighted bar to select **"Runfunc"** and **"Auto (B & P & V [&S])"**. The sum – menu will be displayed as shown.

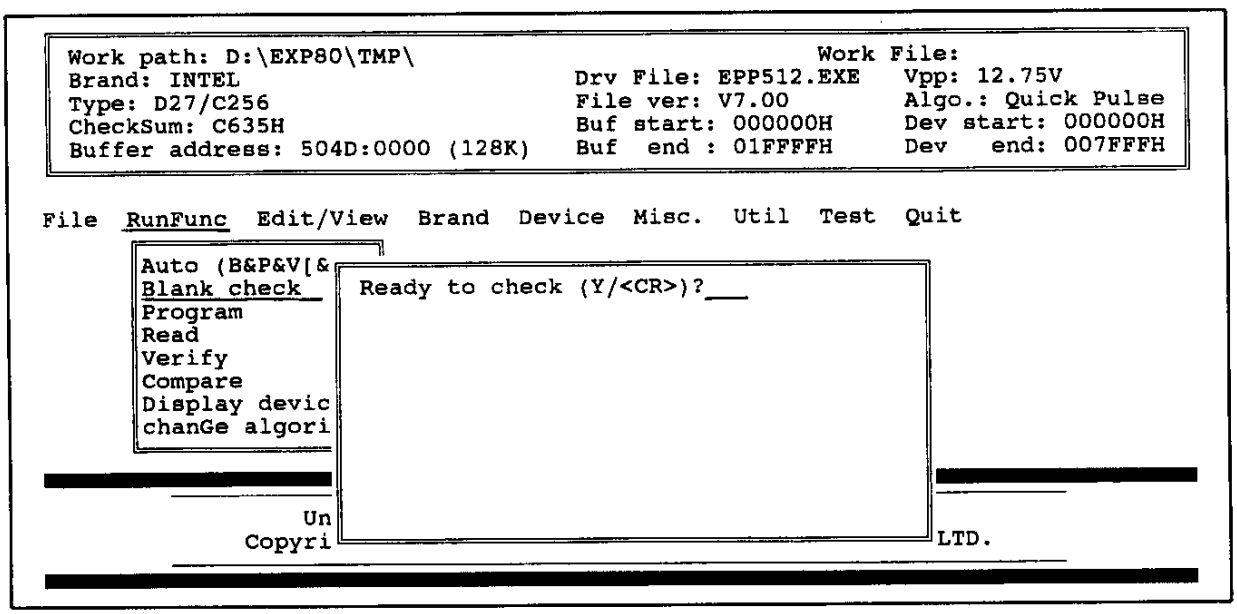


Sub – menu of Auto

You can then press '**Y**' to start Auto function, or <CR> to return to the "Runfunc". This function is almost the same as Program function except that the **blank check**, the **verify** and the **security fuse blown** functions will be **automatically** executed.

4.6 Blank Check

Type '**R**' to select "**Runfunc**" followed by '**B**' to select "**Blank check**"; or move highlighted bar to select "**Runfunc**" and "**Blank check**".
The sum – menu will be displayed as shown.



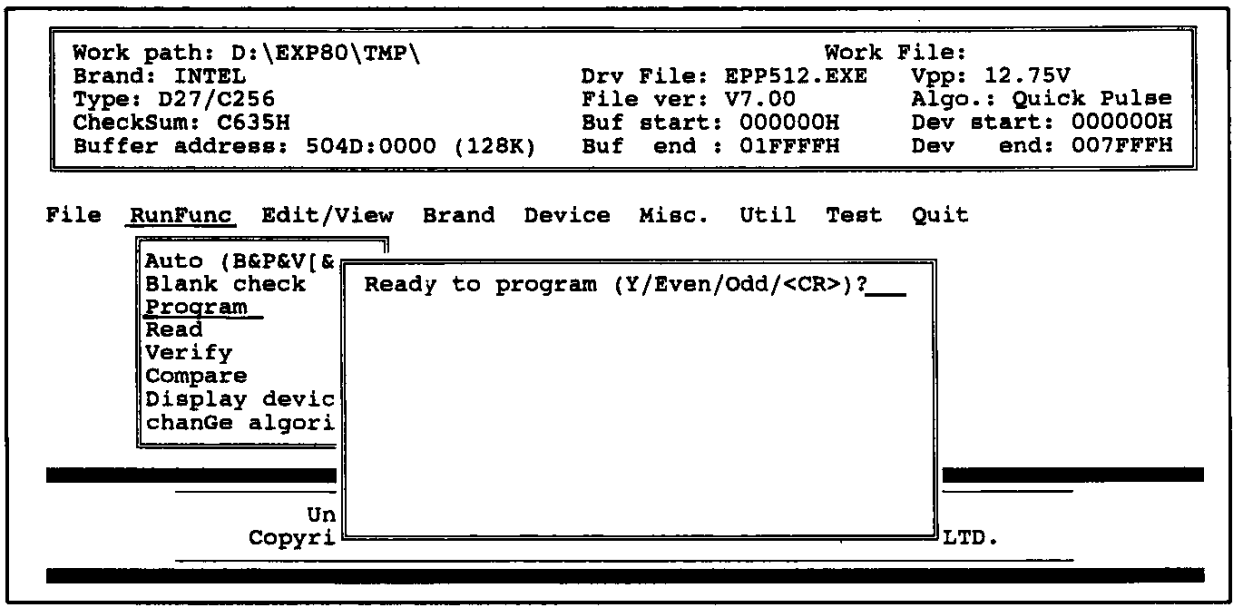
Sub – menu of Blank check

Press '**Y**' to start the blank check, or <CR> to return to the "Runfunc".
If the chip is not blank, the first address will be displayed.

Otherwise, the "**Blank check OK!**" message will be displayed.

4.7 Program

Type '**R**' to select "**Runfunc**" followed by '**P**' to select "**Program**"; or move highlighted bar to select "**Runfunc**" and "**Program**".
The sum－menu will be displayed as shown.



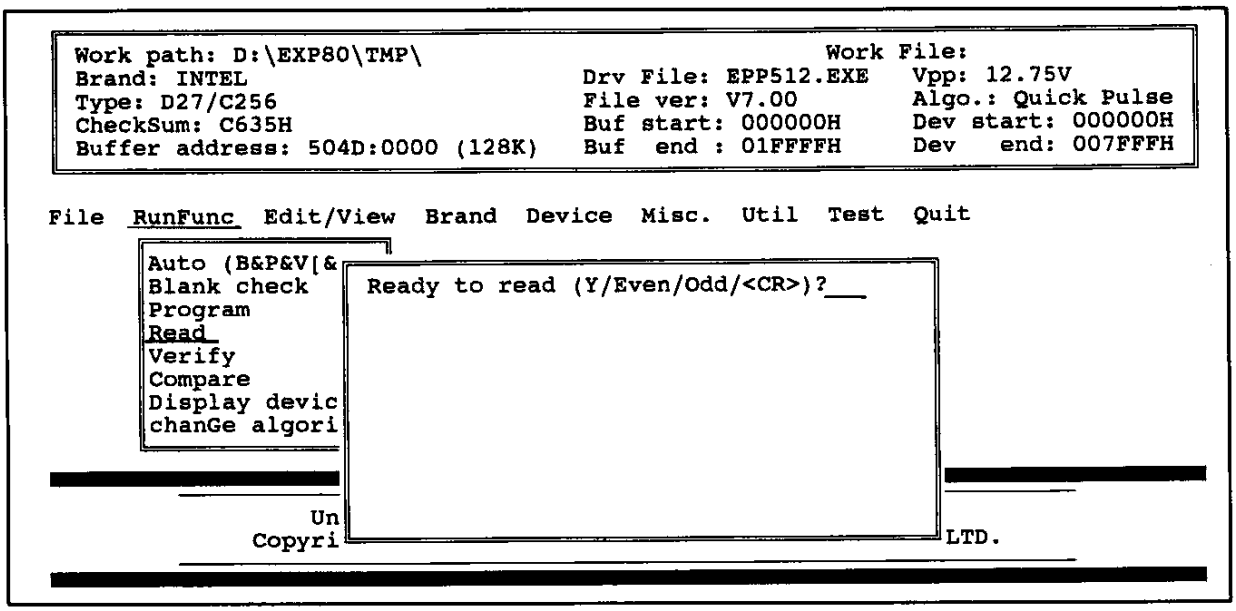
Sub－menu of Program

Now type '**Y**' to begin the transfer of program data from the memory buffer to the DEVICE, or press <CR> to return to the "Runfunc".
The first error address will be displayed when encountering program failure. Otherwise, the "**Program OK!**" message will be displayed on th screen.

After programming is completed it automatically **verifies** the programmed data.

4.8 Read

Type '**R**' to select "**Runfunc**" followed by '**R**' to select "**Read**"; or move highlighted bar to select "**Runfunc**" and "**Read**".
The sum－menu will be displayed as shown.



Sub－menu of Read

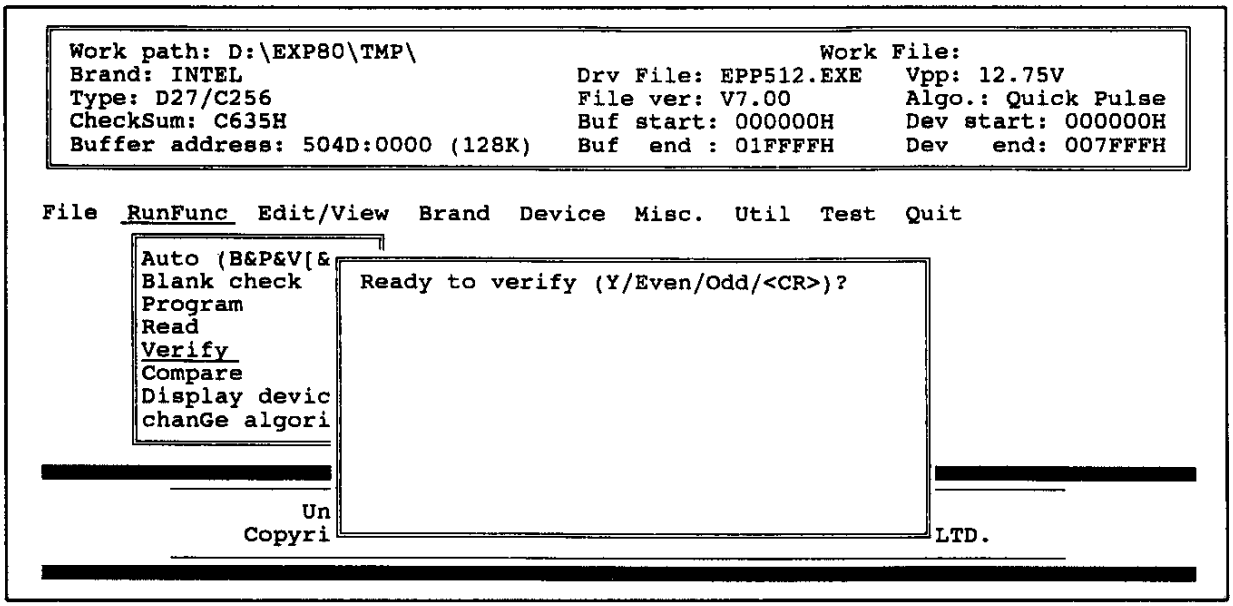
Press '**Y**' to read data from the DEVICE to the buffer, or press <CR> to return to the "Runfunc". The sub－menu will display the "**Reading now...**" message, and when the reading is completed, the "Read OK!" message will be displayed.

The check sum will be automatically calculated after the device

reading.

4.9 Verify

Type '**R**' to select "**Runfunc**" followed by '**V**' to select "**Verify**"; or move highlighted bar to select "**Runfunc**" and "**Verify**".
will be displayed as shown.



Sub – menu of Verify

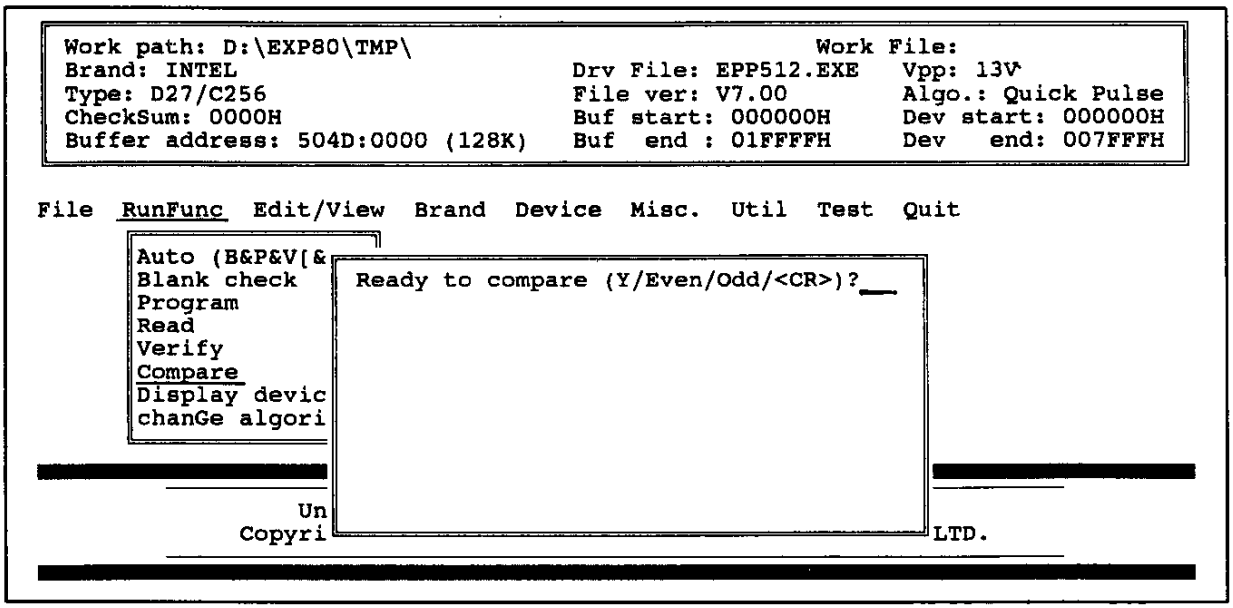
Press '**Y**' to verify the DEVICE with the buffer shown in the **STATUS FIELD**, or press <CR> to return to the "Runfunc". The sub – menu will display "**Verifying Now**" message. When the verifying process is completed, "**Verify OK!**" message will be displayed on the screen.

The verifying routine will be terminated if no error is detected.

4.10 Compare & Display Error

Type '**R**' to select "**Runfunc**" followed by '**C**' to select "**Compare**"; or move highlighted bar to select "Runfunc" and "**Compare**".

The sum – menu will be displayed as shown.



Sub – menu of Compare

NOTE

The programmer will compare the buffer shown in the **STATUS FIELD** with the **chip** in the socket.

Press '**Y**' if you wish to compare the data on the chip with the memory buffer, and the screen will diaplay the differences in the format which follow. Otherwise, press <CR> to return to the "Runfunc".

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 the **'Ctrl + S'** keys simultaneously to hold the display. Press <Esc> to terminate the display, then press any key to return to the "Runfunc".
If no errors are detected, **"Compare OK!"** message will be displayed.

4.11 Display contents of DEVICE

Type **'R'** to select **"Runfunc"** followed by **'D'** to select **"Display device"**; or move highlighted bar to select "Runfunc" and **"Display device"**. The sum - menu will be displayed as shown.

```
Press <ESC> to stop displaying

0000  36 36 31 31 38 38 31 31 --30 30 32 32 38 39 20 20 66118811002289
0010  43 43 4F 4F 50 50 52 52 --2E 2E 20 20 49 49 42 42 CCOOPPRR..  IIBB
0020  4D 4D 20 20 31 31 39 39 --38 38 34 34 FA B4 DD E8 MM  11998844....
0030  60 43 2B C0 8E C0 B9 08 --00 0E 1F BE F3 FE BF 20 `C+............
0040  00 A5 47 47 E2 FB 2B C0 --8E C0 B9 08 00 0E 1F BE ..GG..+.........
0050  23 FF BF C0 01 A5 47 47 --E2 FB 2B C0 8E D8 8E C0 #.....GG..+.....
0060  C7 06 08 00 C3 E2 C7 06 --14 00 54 FF C7 06 62 00 ..........T...b.
0070  00 F6 B0 60 E8 8E 03 B0 --09 E6 60 E8 1F 00 8A F8 ...`......`.....
0080  E8 1A 00 8A E8 8A CF FC --BF 00 05 E4 64 A8 01 74 ............d..t
0090  FA E4 60 AA E6 80 E2 F3 --EA 00 05 00 00 E4 64 A8 ..`...........d.
00A0  01 E1 FA E4 60 C3 FA B4 --D5 9E 73 2A 75 28 7B 26 ....`.....s*u({&
00B0  79 24 9F B1 05 D2 EC 73 --1D B0 40 D0 E0 71 17 32 y$.....s..@..q.2
00C0  E4 9E 76 12 78 10 7A 0E --9F B1 05 D2 EC 72 07 D0 ..v.x.z......r..
00D0  E4 70 03 EB 04 90 E9 D3 --00 B8 40 00 8E D8 E4 64 .p........@....d
00E0  A8 04 75 03 E9 9A 00 B0 --8F E6 70 EB 00 E4 71 86 ..u.......p...q.
00F0  C4 80 FC 09 74 3C 2A C0 --E6 F1 B0 11 E6 20 EB 00 ....t<*...... ..
0100  B0 08 E6 21 EB 00 B0 04 --E6 21 EB 00 B0 01 E6 21 ...!.....!.....!
0110  EB 00 B0 FF E6 21 B0 11 --E6 A0 EB 00 B0 70 E6 A1 .....!.......p..
0120  B0 02 EB 00 E6 A1 EB 00 --B0 01 E6 A1 EB 00 B0 FF ................
0130  E6 A1 B0 8F E6 70 EB 00 --2A C0 E6 71 86 E0 3C 0A .....p..*..q..<.
0140  77 2C BE 58 01 03 F0 03 --F0 2E 8B 1C FA B8 30 00 w,.X..........0.
0150  8E D0 BC 00 01 FB FF E3 --6E 01 B0 09 97 11 4A 11 ........n.....J.
0160  9B 16 71 01 BC 11 9A 11 --F7 07 52 42 7D 01 EB 11 ..q.......RB}...
```

Sub – menu of Display

The screen will display the address and current contents of that address in the DEVICE. Press <**Esc**> to exit the display, and press any key to return to the "Runfunc".

4.12 Copy file to EPROM, Read EPROM to file

As the usable memory of the PC is not greater than 256K bytes, loading a 4M bits (512K bytes) file into the PC buffer can not be performed. The user must therefore split the file into two 256K Bytes files, and load them separately to program a 4M bits EPROM.

Having overcome this problem for 1M EPROM (or over), the user need not worry about the "**memory not enough**" message as appeared in the STATUS FIELD .

Two additional functions can be selected under the "Runfunc" :

1. **Directly copy file to EPROM without going through the PC buffer.**
2. **Directly read EPROM to file without going through the PC buffer.**

F.1 Copy file to EPROM

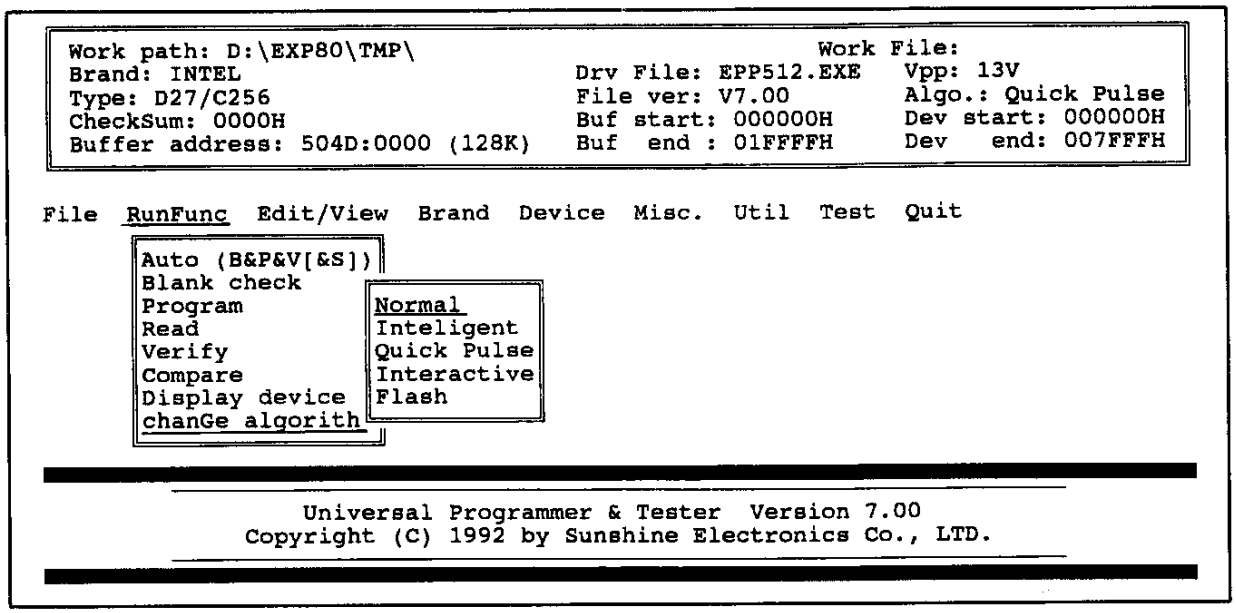
After selecting this function, the software will require you to input the file name. It will then program all data from the **specified** file to the **specified** EPROM on the screen in TYPE status field. Take the file size as basis if it is smaller than EPROTM size and vice versa.

F.2 Read EPROM to file

After selecting this function, the software will require you to input the file name. It will then read all the data from the **EPROM** to the **specified** file on the disk. Take the file size as basis if it is smaller than EPROM size and vice versa.

4.13 Change Program Algorithm

Type '**R**' to select "**Runfunc**" followed by '**G**' to select "**chanGe algorithm**"; or move highlighted bar to select "**Runfunc**" and "**chanGe algorithm**". The sum - menu will be displayed as shown.



Algorithm select

When prompted, key in the number of the new **programming algorithm** you wish to work on. The algorithm you have selected will be updated in the **STATUS FIELD** for programming reference.

Press <Esc> or <CR> to return to the "Runfunc".

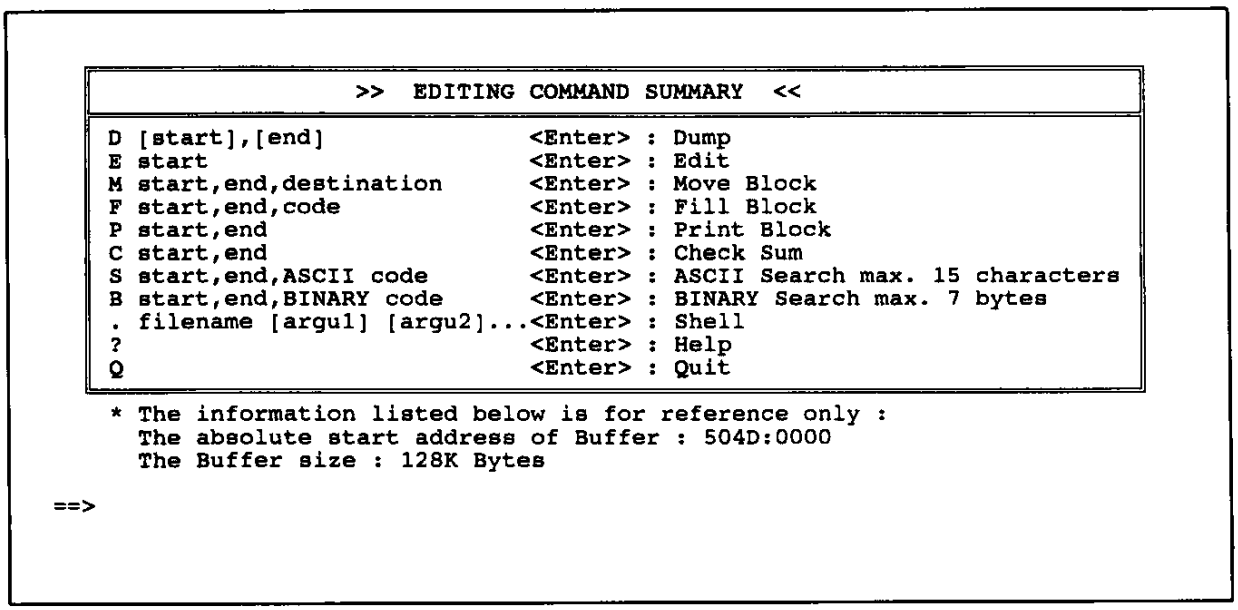
4.14 Security, Lock Bits & Encryption Code

Some MCUs, in addition to ROM code programming function, has other facilities such as **Lock Bits, Security Bits** and **Encryption Code** to prevent unauthorized copy. MCU 8748AH, 49AH and 87C51 series meet this category.
The user can easily operate these functions individually by selecting **"Lock bits", "Security"** or **"Encryption"** under the **"Runfunc"** or select **"Auto"** function to automatically execute the above 3 function. (Each individual

definition is available in your MCU's Manual.)

4.15 Edit Buffer

Type **'E'** to select **"Edit/View"** followed by **'E'** to select **"Edit buffer"** ; or move highlighted bar to select **"Edit/View"** and **"Edit buffer"** to obtain the editing command summary on the screen is as follows:



Edit command summary

You may now proceed with the Edit procedure under command prompt "==>" by using the above command format.

Following the command format, the actual buffer base address is also 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 may pass this address to their own editing program which edit the buffer contents in different ways.

EXAMPLE: The user's editor is FRED.EXE, and he can use the DOT command to pass the base address as follows:

.FRED SEGMENT : OFFSET <CR>

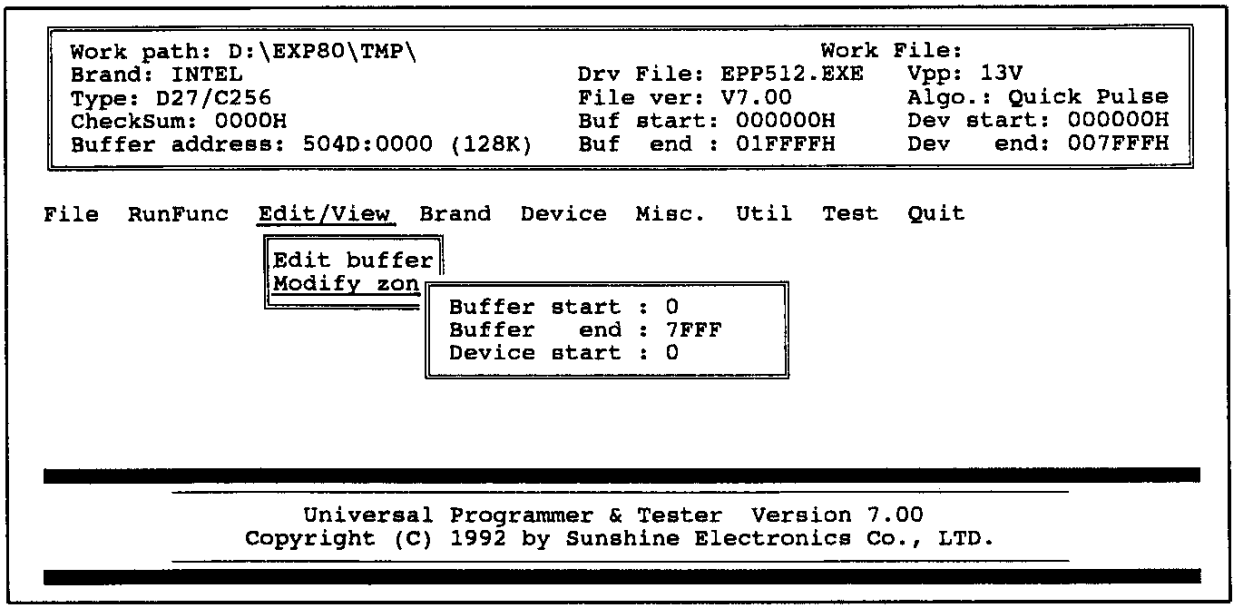
The user must obtain this parameter from the command line of his program as in executing his program under the DOS command prompt:

A:>FRED SEGMENT : OFFSET <CR>

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

4.16 Modify Buffer (Target Zone)

Type '**E**' to select "**Edit/View**" followed by '**M**' to select "**Modify zone**"; or move highlighted bar to select "**Edit/View**" and "**Modify zone**".
The sum – menu will be displayed as shown.



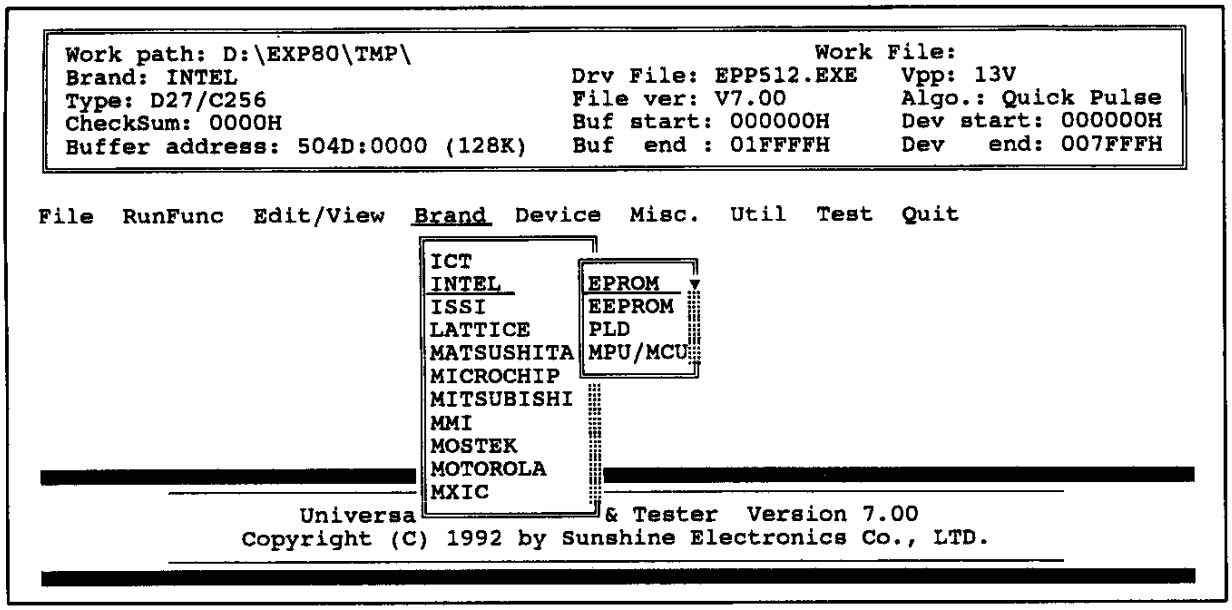
Modify buffer

When prompted, key in the new buffer start address and press <CR>. Next, key in the new buffer end address and press <CR>.
Then, input the new DEVICE start address and press <CR>.
The values you have keyed – in in HEX code will be updated in the **STATUS FIELD** for programming reference.

Press any key to return to the "Edit/View".

4.17 Brand selection

Type **'B'** or move highlighted bar to select **"Brand"**.
The sum – menu will be displayed as shown.

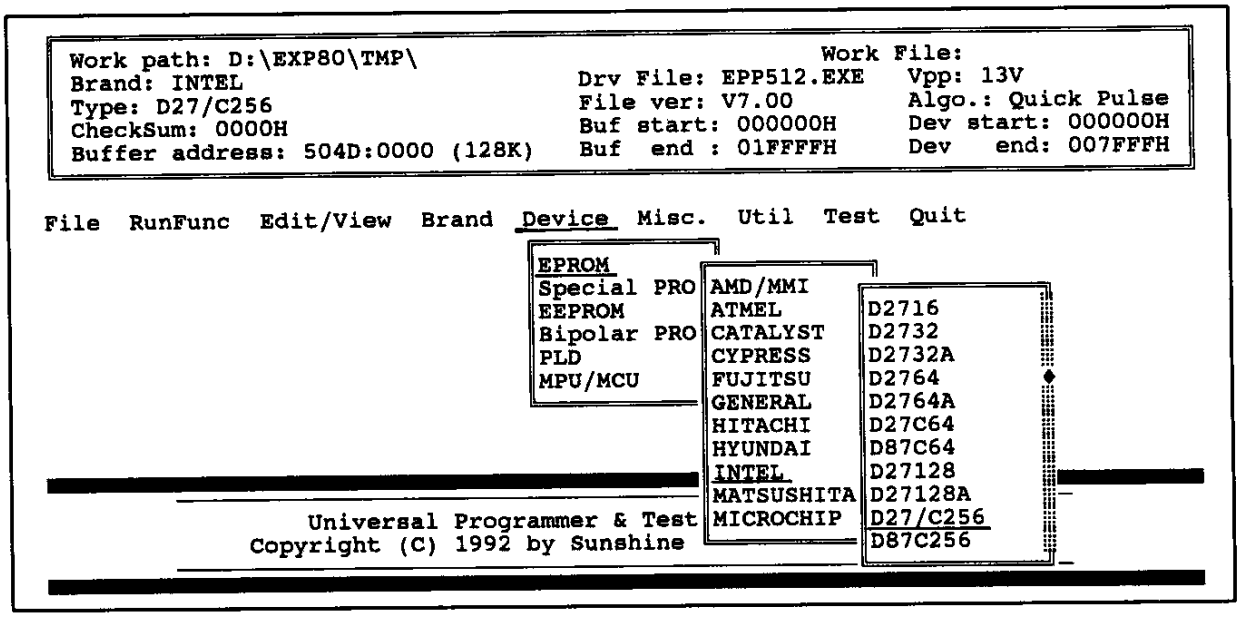


Sub－menu of MFR.

Move the highlighted bar to Select the manufacturer corresponding to your DEVICE. After selecting the Manufacturer, Device and Type, the new **Manufacturer , Type** will be updated in the **STATUS FIELD** for programming reference. It will **automatically** return to the "**Runfunc**".

4.18 Device Selection

Type '**D**' or move highlighted bar to select "**Device**".
The sum－menu will be displayed as shown.



Sub – menu of Device

Move the highlighted bar to select the Device corresponding to your type of Device. After selecting the Device, Manufacturer and Type, the new **Manufacturer, Type** will be updated in the **STATUS FIELD** for programming reference. It will **automatically** return to the "**Runfunc**".

NOTE

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

4. 19 Change I/O Base Address of the System Adapter Card

Before performing this function you must also turn the **DIP SWITCHES ON or OFF** on the system adapter **SAC－101A** card. (This must be done when the computer is switched **off**.)

Please study the following list of I/O address and their corresponding DIP SWITCH positions. **Only one position can be selected ON**.

	DIP SW	POSITION	I/O ADDRESS
	SW1	1	200H
	SW1	2	210H
	SW1	3	220H
	SW1	4	230H
	SW1	5	240H
	SW1	6	250H
	SW1	7	260H
	SW1	8	270H
	SW2	1	280H
	SW2	2	290H
	SW2	3	2A0H
	SW2	4	2B0H
	SW2	5	2C0H
	SW2	6	2D0H
(DEFAULT)	**SW2**	**7**	**2E0H**
	SW2	8	2F0H

Type '**M**' to select "**Misc.**" followed by '**S**' to select "**Set PC I/O addr.**"; or move highlighted bar to select "**Misc.**" and "**Set PC I/O addr.**".
The sum－menu will be displayed as shown.

Then select the I/O address that is set on the system adapter card. The software will automatically check the hardware I/O

circuit and display the result showing **"I/O address is correct"**. If the result appears as **"Communication error"**, it implies the software I/O address does not match the hardware I/O address. Re-check both I/O addresses carefully.

4.20 Quit

Press **'Q'** or **<Esc>** to quit the main menu and return to DOS.
Before quiting to DOS, the parameters of the **manufacturer, type, I/O address, work path** and **programming algoritm** will be saved in the parameter file EXPRO.DAT for later use.

4.21 External Key and LEDs (for EXPRO-80)

The only external key on the programmer module is the **"Yes"** key.
The function of the **"Yes"** key is the same as the **'Y'** on the PC keyboard. When the sub-menu require you to enter **'Y'**, you may press the **"Yes"** key instead of pressing **'Y'** on the PC keyboard.

The three LEDs on the programmer module are **"ON"** LED, **"BUSY"** LED and **"GOOD"** LED.

The **"ON"** LED will light up after switching on the computer power. The **"BUSY"** LED will light up when performing device programming, reading, blanking and vefirying. The **"GOOD"** LED will light up when the verifying result proves errorless.

FOR PLD (PAL, GAL, PEEL, EPLD)

The functions on PLD programming are almost the same as those on the EPROM. The following section describes only the extra functions.

4.22 Extra Functions of PLD Programming

Load JEDEC File:

To program PLD, the user need to load JEDEC Fuse Map file (not Binary file). This is a standard PLD file produced mostly by standard PLD compilers, such as ABEL, CUPL, PALASM II, SNAP, OPAL, PDK－1 and PLAN II. Please refer to **section 4.1** for operating details.

Save JEDEC File:

Under PLD Device file, data is always saved in JEDEC file format for later use. Please refer to **section 4.2** for operating details.

Edit Buffer:

Though the Edit Buffer function is included, we recommend that you use it only for reference. To edit directly on buffer, the user must master the PLD JEDEC Fuse Map assignment very well. Therefore, you are advised to edit with the PLD compiler.

Display Device:

Under PLD Device File, the content of device is displayed in the JEDEC format.

Security Fuses Blow :

The final step in PLD programming is to blow the security fuses. Doing this will prevent any further access into the PLD through "**modefication**" or "**reading**". It also prevents anyone from making unauthorized duplication of your PLD. This function is

automatically executed in the ultimate process of the "**auto**" function.

5. USING THE UTILITY FILES

5.1 HEX to BINARY Code Converter

The converter can convert a HEX format ASCII file to a RAW BINARY file. Some Assemblers or Compilers can produce a HEX format file from a user's source program, then transmit it to a standalone programmer or In-Circuit Emulator through a RS-232C hardware interface.

This programmer module is not stand-alone. It is controlled by the CPU on the PC using a direct I/O control technique; it also shares the large memory resources of a PC. This programmer software will load any type of file without any conversion during the file loading.
It continuousily loads the file content byte by byte into the software buffer for programming purpose.

This HEX converter will convert your **HEX** format file to an executable **ROM code** that is recognizable to your target CPU.

We assume that the user understands the differences between the HEX format file and the ROM code BINARY file.

There are altogether 4 types of HEX format files that can be converted to BINARY files using HEXBIN.EXE with a maximum conversion size of 64K bytes.

1. **INTEL HEX format**
2. **MOTOROLA S HEX format**
3. **TEKTRONIX HEX format** (seldom used)
4. **TI HEX format** (seldom used)

HEXBIN2.EXE can convert INTEL, MOTOROLA, DIGITAL RESEAECH and TEKTRONIX from extended HEX format files to binary files. There is no limit to its maximum conversion size.

The starting address may be specified on HEXBIN2, and the earlier will be omitted in order to keep the binary file in small capacity.

The input command under DOS command prompt is:

A:》 HEXBIN2 [HEX FILE NAME] [BIN FILE NAME] [HEX FORMAT] [start address] <CR>

[]:Option <CR>:return key or enter key.

HEX FILE NAME and BIN FILE NAME are standard file name that are specified by DOS.

HEX FORMAT: **I** for INTEL HEX
M for MOTOROLA HEX
D for DIGITAL RESEAECH HEX
T for TEKTRONIX HEX

Start address: HEXADECIMAL digit, this parameter need not be assigned in the HEXBIN converter.

1. In the **INTEL** extended HEX format, the starting address represents the starting segment address. All data lying between the starting segment x 16 and the end of the file will be picked up. The segment range is from 0H to F000H.

2. In the **MOTOROLA** S HEX format, the starting address represents the actual starting address. All data lying between the starting address and the end of the file will be picked up. The address range is from 0H to FFFFFFFFH.

For example:

A:》HEXBIN2 <CR>

HEX FILE NAME : DEMO.HEX
BIN FILE NAME : DEMO.BIN
HEX FORMAT
<**I**>INTEL/<**M**>MOTOROLA/<**D**>DIGITAL/<**T**>
TEKTRONIX:I
SEGMENT ADDRESS:1000
TO FILL UNUSED LOCATION (<**0**> 00/<**1**> FF):1

5.2 Dump BINARY file to Console

Most (ROM code) **BINARY** files cannot be displayed on the screen by a DOS TYPE command. DUMP.EXE can convert a **BINARY** file to **HEXADECIMAL** characters and display them on the console or printer. Although it is meaningless, a user may like to keep a copy for later use.

The input command under DOS command prompt is:

A:》[^P]DUMP FILENAME [start address] <CR>

[]:Option <CR>:Return key or Enter key
[^P]:**Ctrl+P**

It will connect the PC with the printer, and print out the data displayed on the screen.

FILENAME:standard file name specified by DOS.
Start address: HEXADECIMAL digits start dumping from this address, range from 0H to FFFFFH.

5.3 2-way, 4-way BINARY File Splitter

SPLIT2.EXE:
One file compiles the data from low byte of the **16－bit** file and can program it to an **EVEN** EPROM.
Another file compiles the data from high byte and can program it to an **ODD** EPROM.

SPLIT4.EXE:
The first file collects data from the 1st byte of the **32－bit** file, the second file from the 2nd byte and accordingly for the remainding two files.

The input command under DOS command prompt is:

A:》SPLIT2 [input file] [output EVEN file] [output ODD file] <CR>
A:》SPLIT4 [input file] [output 1st file] [output 2nd file][output 3rd file] [output 4th file] <CR>

[]:Option <CR>:Return key or Enter key

input file,
output **EVEN** file,
output **ODD** file,
output **1st** file,
output **2nd** file,
output **3rd** file,
output **4th** file,

There are standard filenames specified by DOS.

5.4 2－way, 4－way BINARY File Shuffle

SHUFF2.EXE can shuffle two 8－bit source files into a **16－bit** file.
The first **8－bit** file will be collected to the **LOW** byte of the

16－bit file, and the second to the **HI** byte.

SHUFF4.EXE can shuffle four **8－bit** source file into a **32－bit** file.
The first **8－bit** file will be collected to the 1st byte of the **32－bit** file, the **second** to the **2nd** byte and so on for the remaining two 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] <CR >

[]:Option <CR>:Return key or Enter key

output file,
input **EVEN** file,
input **ODD** file,
input **1st** file,
input **2nd** file,
input **3rd** file,
input **4th** file,

These are standard filenames specified by DOS.

5.5 Edit a BINARY File

EDBIN.EXE can edit a BINARY file. There is **no limit** on buffer size.
The editing command summary on the screen is as follows:

```
                    SUNSHINE Electronics Co.,Ltd.
                Utility for edit BIN file  V1.01 (C)1992

                          << 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
W filename,start,end     <RETURN> : WRITE BLOCK TO FILE
A filename(second),filename(output) <RETURN> : APPEND FILE
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
Q                        <RETURN> : QUIT
=====================================================================
>Current file name : dump.exe
>File length : 9356 bytes

==
```

Edit command summary

You may now proceed with the Edit procedure under command prompt "==" by using the above command format.

A:》EDBIN [input filename]

[]:Option <CR>:Return key or Enter key

5.6 Separate a BINARY File large than 2Mbits (256K bytes)

SEP2MB.EXE will separate the source file into two files, the first file being always 2Mbits(256K byte) and the second file the remainder.

A:》SEP2MB [input filename] [output filename1] [output filename2]

[ ]:Option <CR>:Return key or Enter key

APPENDIX A.

IC TESTER FUNCTION DESCRIPTION

1. Specifications

1.1 Applicable IC type

74/54 TTL, HC, HCT or equivalent CMOS series
40/140 CMOS series
45/140 CMOS series
DRAM 4164, 41256, 411000, 4416, 4464, 44256, 4Mx1, 1Mx4
SRAM 6116, 6264, 62256, 1Mx8, 4Mx8 series

1.2 Test Options

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

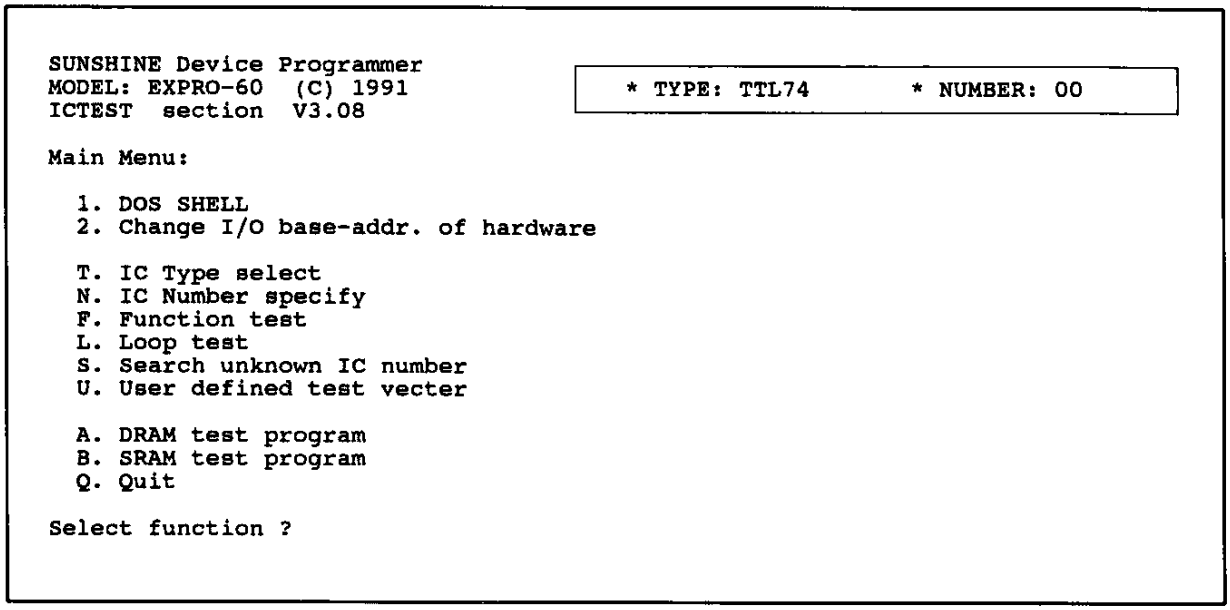
1.3 Files Included on the Software Disk

ICTEST.EXE	:Main program file.
ICTEST.DAT	:Setup and parameter data file.
README.DOC	: Any revision or amendments to the IC test card of software since the printing of this manual are listed in this file.
TTL74.LIB	:TTL library.
CMOS40.LIB	:CMOS40 library.
CMOS45.LIB	:CMOS45 library.
7406.VEC	:Sample vector file.
4040.VEC	:Sample vector file.

1.4 Basic Operation

Type '**T**' to select "**Test**" followed by '**I**' to select "**IC Test**"; or move highlighted bar to select "**Test**" and "**IC Test**" under the main menu.

A dialog window will be displayed on the screen as follows:



IC test

The ICT TEST menu will be displayed. Proceed with testing by selecting the desired function number.

A detailed description of each function is given in Section 2.

2. Function Description

To select a particular function press the **first letter** of that function.

1. DOS Shell:

Please refer to 4.4

T. IC Type selection:

There are 3 IC types for selection, **TTL74, CMOS40 and**

CMOS45.
Press the desired number and the corresponding library will be selected for testing.

N.IC NUMBER specification:
Enter the IC number to be tested, e.g, 174(ENTER). Do not enter the IC type, as this is selected by function **'T'**. The number entered will be checked with the standard library. If the number is not available, you will have to write your own test vectors.
See function **'U'** below.

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

L.Loop test:
This option will repeatedly test the chip logic combinations in an endless loop. The test will stop, then convey an error message in the event of encountering any error during the process. The test may be terminated by pressing any button.

S.Search for unknown IC identification number:
There are 3 libraries for selection. The **unknown** IC will be first checked with the current standard library. If it fails to search the IC, a message will be displayed. You will have to run the other 2 libraries to search. When the unknown IC encounters other ICs of similar logic during the process of searching, the screen will display these numbers.

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 box.

You may write your own vectors to test **new IC's** logic in sequence (including **PAL, EPLA, PROM** etc.). The test vector syntax is described in detail in section 3 (**NOTE**: Only logic testing is carried out.
Loading and IC speed tests are excluded.)

A. DRAM test program:
Pressing **'A'** will produce the DRAM test Menu.

N. DRAM number selection:
Allows the selection of the following 4164, 41256, 411000, 4416, 4464, 44256, 4Mx1, 1Mx4.

F. DRAM function test:
All bits on the DRAM will be written and read for comparison.

B. SRAM test program:
Pressing **'B'** will produce the SRAM test Menu.

N. SRAM number selection:
Allows the selection of the following chips 6116, 6264, 62256, 1Mx8, 4Mx8.

F. SRAM function test:
All bits on the SRAM will be written and read for comparison.

Q. Quit:
Quit to main menu.

3. User Defined Test Vectors

3.1 A test vector is the input and output logic states applicable to the various pins of an IC. A single IC can be edited up to 1500 vectors, allowing test of logic in sequence. It can also test vectors of below 24 pins.

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 1 2 2 2 2 2
Vector Pin No 1 2 3 4 5 6 7 8 9 01 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>

V0001 specifies vector line number

G on pin 12 applies GND to IC.

E on pin 24 applies 5v to IC.

X on pins 1, 2, 3, 22, do not care about input or expected output from IC.

N on pin 6 applied input and expected output are same as the last vector.

O 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 expected output from IC is low.

H on pins 8, 17, 19 expected output from IC is HI.

Note:

In an example where a 14 pin IC is to be tested, the originally sixth pin position in 24 pin IC now becomes the first pin in a 14 pin IC.

3.2 A vector can be executed by selecting the Function or Loop test. The test sequences are as follows:

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 if with the expected value 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.

3.3 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**

3.4 The vector test can be edited by using the built soltware by pressing key **'4'** under USER TEST VECTOR MENU. The editing key functions will be shown on the screen. Other word processors may be used provided they are in ASCII code, e. g, Wordstar in the N mode.

3.5 '3' saves the vector to the disk.

3.6 '2' loads the vector into the buffer.

3.7 USER function test and LOOP test.

'F' for vector function testing.
'L' for vector loop testing.

3.8 'D' will invoke the DISPLAY RESULT DURING TESTING function.
This function can display the test results step by step or successively depending on your choice. The results can be printed

This operates in a manner similar to a logic function debugger.

TTL74.LIB

0000 0001 0002 0003 0004 0005 0006 0007 0008 0009
0010 0011 0012 0013 0014 0015 0016 0017 0020 0021
0022 0025 0026 0027 0028 0030 0032 0033 0037 0038
0040 0042 0043 0045 0046 0047 0048 0050 0051 0053
0054 0055 0060 0061 0064 0070 0072 0074 0085 0086
0089 0095 0107 0108 0109 0110 0111 0112 0113 0114
0116 0125 0126 1028 0132 0133 0136 0137 0138 0139
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 0183 0189 0190 0191
0192 0193 0194 0195 0240 0241 0242 0243 0244 0245
0247 0248 0249 0251 0253 0256 0257 0258 0259 0260
0266 0273 0276 0279 0280 0283 0290 0293 0295 0298
0299 0322 0323 0351 0352 0353 0365 0366 0367 0368
0373 0374 0375 0377 0378 0386 0390 0393 0465 0490
0540 0541 0573 0574 0590 0640 0641 0643 0644 0645
0646 0648 0669 0670 0688 0804 0805 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 4464 41464 414256
44256 4Mx1 1Mx4

SRAM

6116 6264 62256 1Mx8 4Mx8

APPENDIX B.

PLD VECTOR TESTER FUNCTION DESCRIPTION

PLD VECTOR TEST

The **PLD** vector test (PALTEST.EXE) presentment will appear after selecting this function. Through this function, the user can test PLD devices using **JEDEC TEST VECTOR** in the JEDEC fuse map file.
A PLD whose security fuses are blown can then be tested easily. The operating procedure is the same as the **ICTEST** function. 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
X: Don't care
C: Apply positive clock
K: Apply negative clock

Exception : The **maximum** number of vectors : **1500**

NOTE:

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

APPENDIX C.

TROUBLE SHOOTING

We have provided the following troubleshooting guide to help you tackle some most **commonly encountered problems**. This guide, however, is not intended to be a repair manual. If you encounter problems other than those described here, please contact your dealer or our Sales Department.

INSTALLATION PROBLEMS

PROBLEM1 :
When I turn my computer on, I hear no beeps, the fan doesn't spin, nothing happens!

RESOLUTION 1 :
1 - 1 The power cord may be disconnected from the computer or the wall. Check the power cable.
1 - 2 You may not have a chip correctly inserted in the ZIF socket. Make sure that your chip is correctly installed and the handle is down.
1 - 3 Your power supply is insufficient to drive both your system and system adapter card.

PROBLEM 2 :
When I try to use a programmer module, I get communication error messages!

RESOLUTION 2 :
2 - 1 You may not have the I/O port set correctly for your programmer. Double check the I/O port assignation.
2 - 2 You may not have a chip correctly inserted in the ZIF socket. Make sure your chip is correctly installed and the handle is down.
2 - 3 It might be due to improper connection between the system

adapter card and the programmer module. Double check the cable connection.

2－4 Your system may be running too fast. Try to slow your system down to the greatest possible extent, or try to use an IBM AT－8 MHz or compatible.

2－5 The bus speed on your system may be too fast. The system adapter card will not run with bus speed greater than 386 25 MHz.

PROBLEM 3 :

When I install the system adapter card, some of my other peripherals have strange reactions!

RESOLUTION 3 :

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

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, according to the instructions in this manual.
4. Make sure that all cards and cables are firmly attached.
5. Remove any memory resident programs from memory.
6. Check whether 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 in above above section, check the following :

1. Is the system adapter card fully inserted in its slot ?
2. Are all cable connections securely attached ?
3. Does the system adapter card jumper setting match the I/O address displayed by the programmer software ?
4. Does any other card on the bus has the same I/O address as the system adapter card ?