USER'S MANUAL
SEP-81AE/84AE
EPP-01AE/04AE/01CE

PC-BASED
QUICK EPROM PROGRAMMER

HI-LO SYSTEMS

USER'S MANUAL

SEP/EPP Series

PC-Based EPROM Programmer

Version V1.1
October 1994

HI-LO SYSTEMS

Model numbers mentioned in this manual are:

System Adapter Card : SAC-201B
Programmer Module : SEP-81AE, 8 Mbit, 1 socket
SEP-84AE, 8 Mbit, 4 sockets
EPP-01AE, 1 Mbit, 1 socket
EPP-04AE, 1 Mbit, 4 sockets
EPP-01CE, 512 Kbit, 1 socket

CONTENTS

APPENDICES

1. INTRODUCTION

1.1 Manual Contents

This manual describes the methods for installing and operating the EPROM Programmer with an IBM PC or compatible running MS-DOS or PC-DOS.

A user, who is familiar with general PC software installation problems, can easily prepare the equipment for operation.

This manual also contains information about the EPROM Programmer's usage and detailed functions.

1.2 Product Configuration

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

* PC System Adapter Card (SAC-201B).
* 1-M. cable with two 25-pin, D-type connectors on each of the cable's ends.
* Programmer module.
* User's manual.
* One diskette with the following files:

README.DOC:	Software & hardware revision message.
EP8x.EXE:	Main program for the programmer module.
EP8x.DAT:	Initializing parameters which will be loaded upon main program execution. (x: 1 for one-socket programmer module, 4 for four-socket programmer module.)
DUMP.EXE:	Dump file to console in binary format.
HEXBIN.EXE:	HEX to BINARY file converter.
HEXBIN2.EXE:	EXTENDED HEX to BINARY file converter.
SPLIT2.EXE:	2-way file splitter.
SPLIT4.EXE:	4-way file splitter.
SHUFF2.EXE:	2-way file shuffler.
SHUFF4.EXE:	4-way file shuffler.

2. INSTALLATION

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

The installer can easily set up the programmer module for service by checking the system requirements and performing the following steps:

2.1 Host System Requirements:

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

2.2 Software Installation Procedures:

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

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

Steps as follows:

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

Whenever a new updated diskette reaches you, by applying the above mentioned method, you can easily perform your own updating.

Then, proceed with the Hardware Installation Procedure.

2.2.2 The PC system has floppy disk drive only.

Proceed with step 2.3 Hardware Installation Procedures, because this configuration needs no Software Installation.

2.3 Hardware Installation Procedures:

Before starting the Installation Procedures, it is necessary for the user to make and use a work-copy disk of the original software diskette. 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 main program. To install the EPROM Programmer and the software supplied, follow these steps:

Step 1:
Switch off your computer system, and carefully open the computer cover.

Step 2:
Check the switches of the PC system adapter card,
For I/O address selection 2E0H (default):
SW1 : only position 2E0 should be fixed with the jumper.

Step 3:
Gently insert the system adapter card 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 connected to the system adapter card; the female end must be connected to the programmer module.

CAUTION
======

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

Step 5:
Turn on the computer and check the LEDs on the programmer module.

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

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

Step 6:
Boot DOS and then perform "Change I/O Base Address" function according to the following sequences:

Step 6-1
Boot DOS in disk drive a:
In response to the operating system prompt, replace the DOS diskette with a work copy and type the following command, if you are using the main program:

A>EP8x <CR>

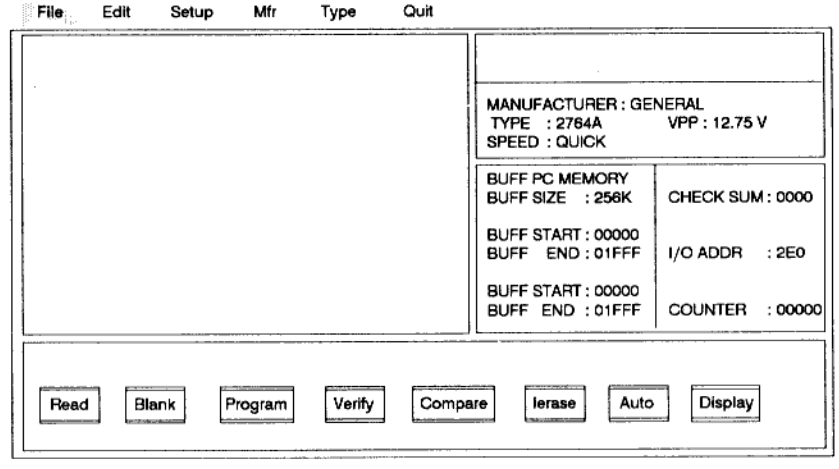
<CR> is the RETURN or ENTER key.

NOTE

x stands for socket number of the programmer. 1 for the one-socket programmer module and 4 for the four-socket programmer module.

The main program will be executed, and will look for the parameter file EP8x.DAT. If it exists on the default disk drive, the parameters of I/O address, Manufacturer, Type and Programming Speed will be loaded as your default state. Otherwise the values set before delivery, I/O address (2E0), Manufacturer (General EPROM), Type (2764A), Programming Speed (intelligent) will be considered as the default state.

At the same time, the main program will also check the identification circuit in the programmer module. Once it has been checked, the main program will display the following function menu on the screen and wait for the user's command.



Main menu

Step 6-2
If the function menu is not displayed on the screen, the work-copy diskette is possibly bad. Remake and use another work-copy diskette and then retry step 6-1.

NOTE

If this problem persists, please contact your local distributor for help.

Step 6-3
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 system adapter card is mismatched with the software default I/O address, or conflicts with other add-on cards inserted in the PC. Press "Q" to quit to DOS prompt, turn off the computer and recheck 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 rechecking all these connections, repeat the installation procedure from Step 1.

NOTE

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

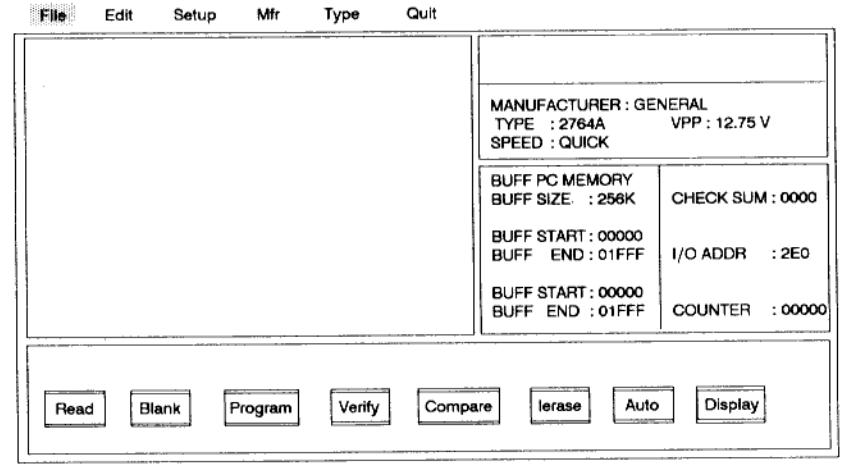
You may also press the RETURN or ENTER key to enter the main menu, then select the change I/O address under the Setup function group in order to change the I/O address of the system adapter card to other available addresses on the PC.

3. USING THE PROGRAMMER

In this chapter we will help the user step by step to become familiar with his new EPROM Programmer tools. The Installation Procedures described in the previous chapter must have already been performed.

3.1 Viewing the Main Menu

Studying the main menu before starting to operate the functions will help you familiarize yourself with the window menu features.



Main menu

3.1.1 Main Function Line

The top line of the main menu is called the main function line. It has six groups of functions: File, Edit, Setup, Mfr, Type, Quit.

3.1.2 Immediate Execution Line

The bottom line on the main menu is called the immediate execution line. From left to right, 8 functions for selection are READ, BLANK, PROGRAM, VERIFY,

COMPARE, AUTO, IERASE and DISPLAY; they can be immediately executed through the main program.

It is very simple to select an individual function by pressing the first letter of that function. After pressing the first letter of the desired function, the subwindow or description of that function will be prompted on the screen for the next instruction.

3.1.3 Status Field

On the right side of the main menu there are 4 windows with information on EPROM, from top to bottom, logo, version, current EPROM, Mfr., Type, VPP, Speed, working buffer and EPROM addresses, I/O address... This group of windows is called the status field.

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

3.2 Getting Started

This section presents a simple example to help you become familiar with the programmer module and some commonly used functions. Experiment with the suggested options. If you do not get the desired results, make sure all the cables are connected firmly.

Follow the steps to learn how to start and to exit the main menu and to use functions such as Mfr., TYPE, LOAD, BLANK and PROGRAM. The example here is to practise with a 2764A 8K EPROM; other 27 series EPROMs can also be used for exercise.

Now, let's practise with the 2764A EPROM you have at hand.

You can apply either of the following methods to begin the EPROM programming according the system you are using.

A. If you have copied the main program files into hard disk (see Section 2.2.1),

Change the current directory to the directory under which the main program has been copied. Using EP as an example, execute the main program under EP directory as follows:

C:\EP\EP8x <ENTER>

NOTE

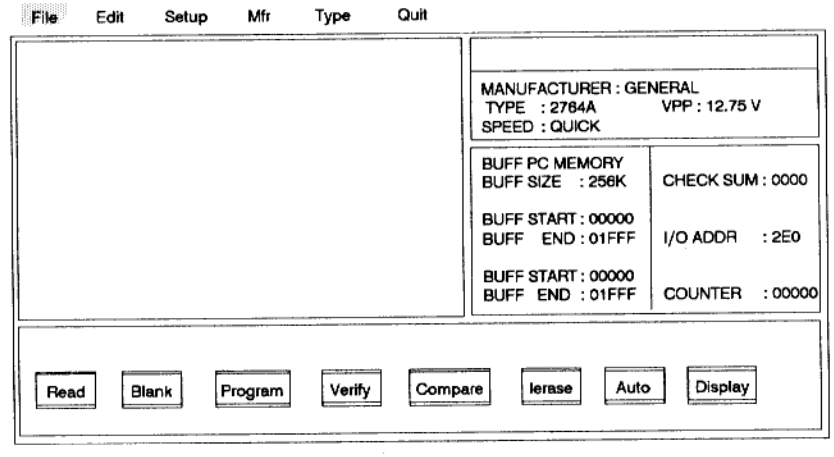
x stands for socket number of the programmer module. 1 for the one-socket programmer module and 4 for the four-socket programmer modules.

B. If you use a floppy disk only,

Your first step is to execute the main program directly from drive A: by typing:

A:>EP8x <ENTER>

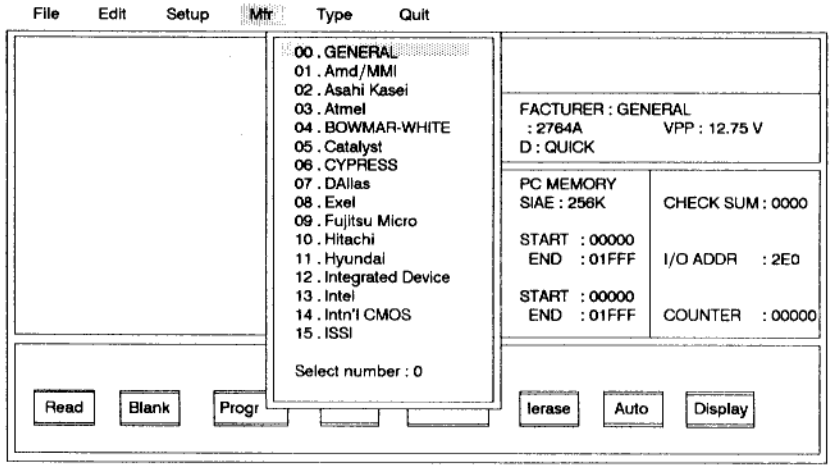
The user can access the following main menu on the screen after proceeding with above method A or B to carry on the programming operation.



Main menu

3.2.1 Select EPROM Type

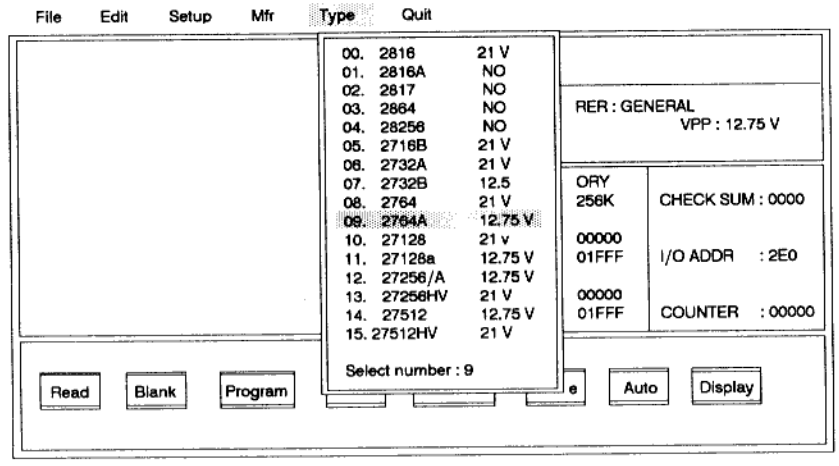
Selecting the EPROM manufacturer, type and voltage should always be the first step. After typing M from the main menu, the screen will display the submenu under Mfr.



Submenu of Manufacturer

Enter the number which indicates a manufacturer corresponding to your EPROM, for example, 00 for general EPROM. The manufacturer will be updated in the status field. The screen will then display the subwindow with supported types for the selected manufacturer.

Then select a number with the EPROM type subwindow that corresponds to the one you are using, for example, 09 for 2764A. The status field will automatically update and display the type and its required voltage as follows:



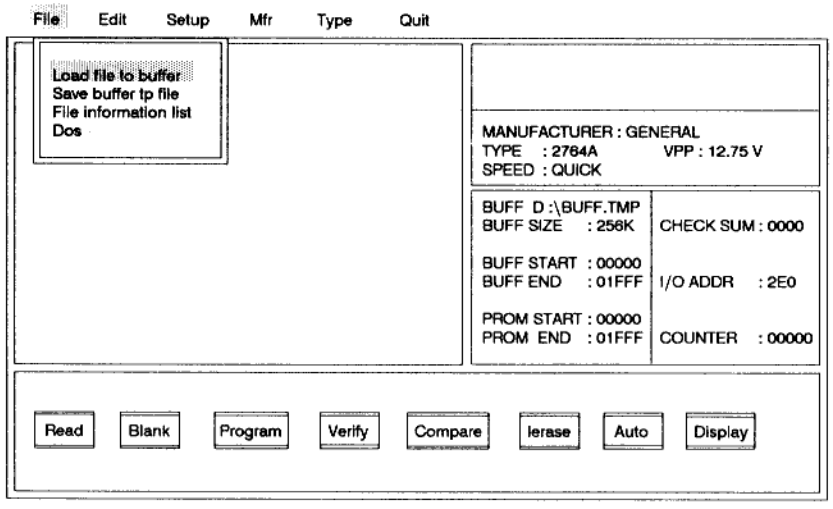
Submenu of Type

NOTE

As EPROM chips may not always be clearly labelled, it would be difficult, for example, to distinguish a 2764 requiring 21V from a 2764A requiring 12.5V. 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. Please refer to Appendix B for more information about EPROM manufacturers, types and voltage selections.

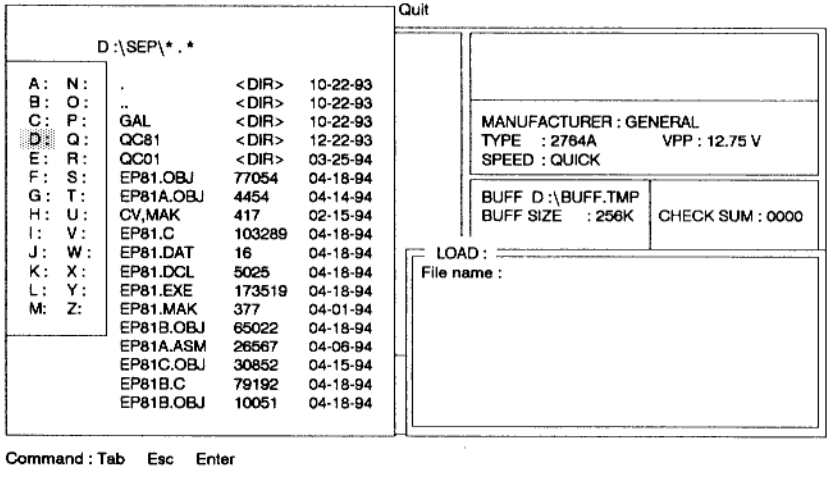
3.2.2 Load Disk File into Buffer

After you have taken the above-mentioned steps, the next step is to transfer the disk file in raw binary format to the memory buffer. Press F to have the File subfunction menu displayed on the screen.



File

Press L, or use the up/down arrow keys to select Load file to buffer function. Press <CR> to access a dialog window for the File loading on the screen.



Load file

NOTE

Files in following three formats can be loaded to the working buffer:

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 and use the following commands:

* <TAB> : switches control between the left and the right windows.

* <UP ARROW> & <DOWN ARROW> :
  used to move the highlighted bar to a file or a subdirectory where a file is 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 NAME> : selects which drive's files will be shown in the selection window on the left.

NOTE:
These commands will only work when the TAB key is pressed and the left window is active/highlighted.

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 best to use this option. Files already in binary format load quicker to the programmer. All HEX files are actually converted to Binary by our software before being loaded to the buffer.

If 'B' is chosen you will be prompted "Load Adress(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 enter in the HEX number of the address where you would like the file to be loaded. The file will now be loaded to the buffer.

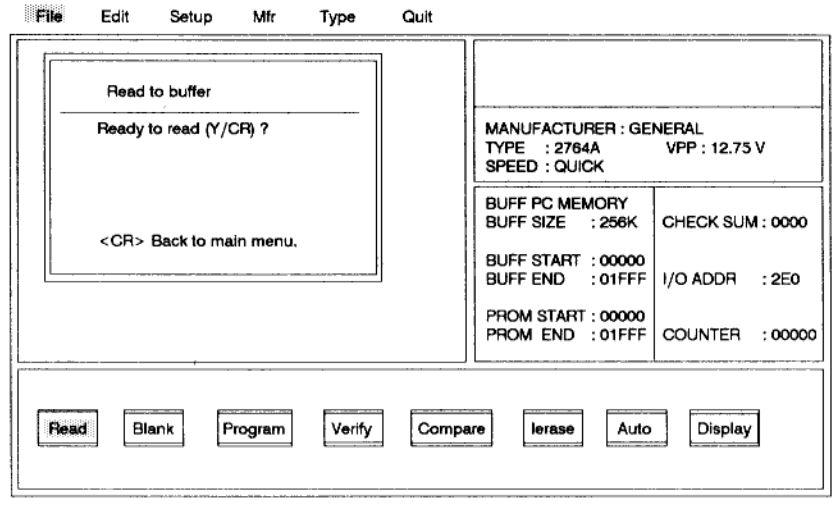
If 'I' is chosen you will be prompted for "File start seg.(0000):-, here the file start segment address is required. All data lying between the start segment X 16 and the end of the file will be converted to binary and loaded to the buffer memory. The segment range is from 0 to F000.

If 'M' is chosen you will be prompted for "File start addr.(000000):-", here the actual start address of the file is required. All data lying between the start address that is given and the end of the file will be converted to binary and loaded to the buffer. The address range is from 0 to FFFFFF.

If 'I' or 'M' was selected above 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 unsed 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 EPROM can be filled with whatever other data you want by loading other HEX files.

3.2.3 Read Contents from Master EPROM

If the EPROM data is in a master EPROM instead of a disk file, you have to enter the READ function by pressing R. The screen will display the subwindow of READ.



Subwindow of Read

Insert the master EPROM into the socket labeled MASTER while the socket lever is up. Correctly set the EPROM GND pin to match the lower left pin on the socket, then push down the socket lever.

Press Y to start the data transferring from the Master EPROM to the memory buffer as specified in the status field. The screen will display:

Reading now...
OK!

After the read operation has been completed, the check sum calculated during the operation will also be displayed in the status field for reference.

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

3.2.4 Insert the Blank EPROM into the ZIF socket

After transferring the data from the disk file or MASTER EPROM to the memory buffer, contiguously insert the blank EPROMs into the sockets (or sockets

labeled from 1 to 4 in the same manner as the master EPROM was inserted into its socket.

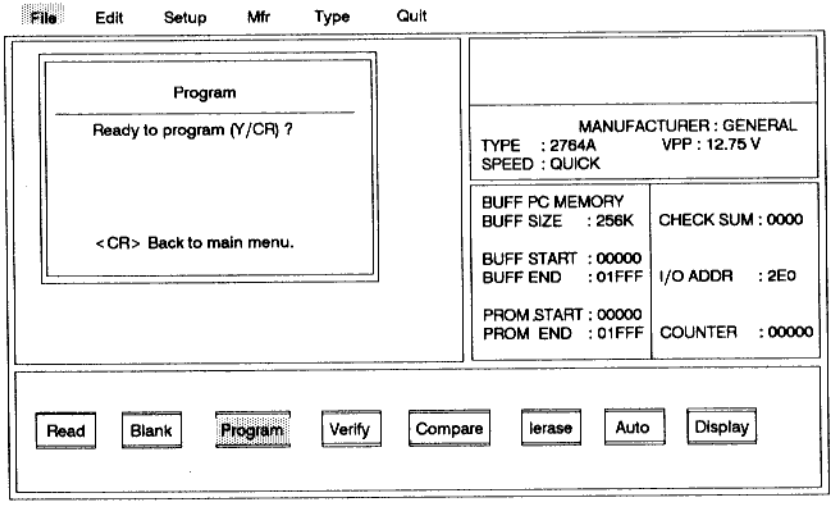
CAUTION

= = = = = =

The EPROM must be inserted with the notched end towards the end of the socket; it will otherwise be destroyed or programmed to an unknown state.

3.2.5 Program Buffer Contents to EPROM

After loading the disk file or reading the master EPROM data into the memory buffer, and inserting the blank EPROMs, press P to program the EPROMs. The subwindow of PROGRAM will appear on the screen.



Subwindow of Program

Press Y to activate the programming function. By the end of the programming process, a verify function with the EPROM contents and memory buffer will be executed automatically. If there are discrepancies, they will be displayed as follows:

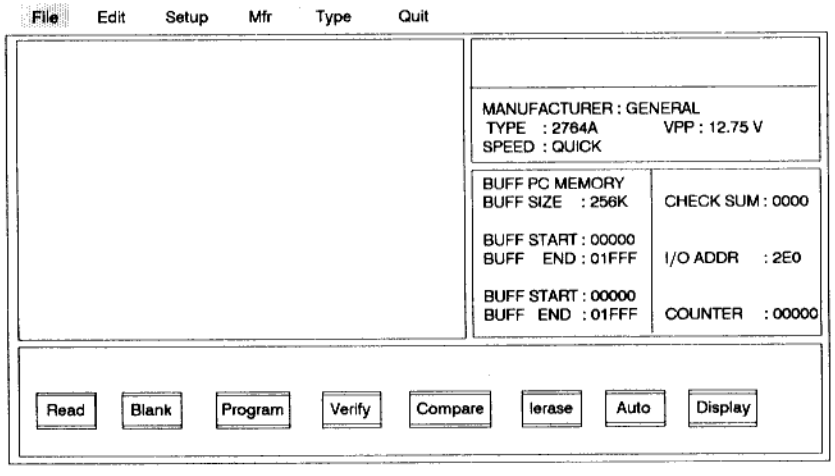
#1. OK! (/ERROR AT XXXX)

This completes the whole programming process. To program other EPROMs, wait till the BUSY LED turns off, then replace the EPROMs and type Y again.

Press <ESC> or <CR> to return to the main menu and exit the programming process.

4. DETAILED FUNCTION DESCRIPTION

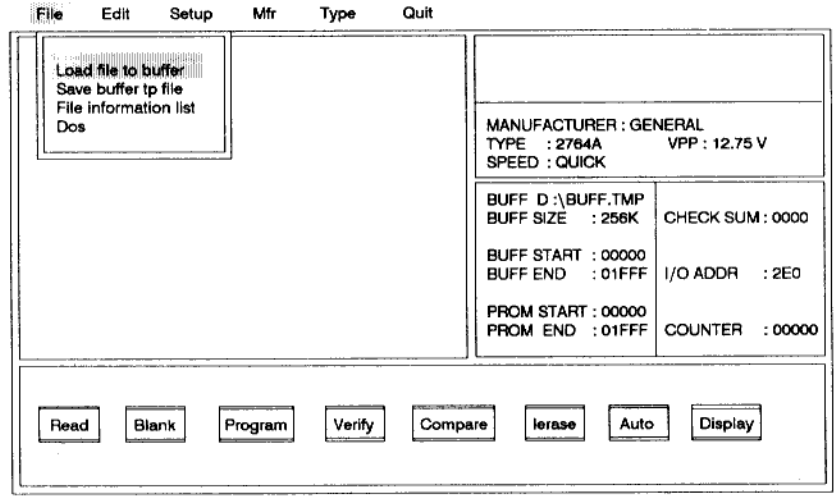
The following detailed function descriptions are illustrated in the order given in the main menu. The functions have two main modes: main function and immediate execution function. The main function has six function groups, File, Edit, Setup, Mfr, Type and Quit, each with several subfunctions; the immediate function is directly associated to programming operation, such as Read, Blank.... Press the first letter of the desired function from the main menu, the screen will immediately display the submenu of that function.



Main menu

4.1 File

To enter **File**, press **F** or use the left/right arrows keys to move the highlighted bar to the desired function and then press <CR>. The File subwindow will appear on the screen.

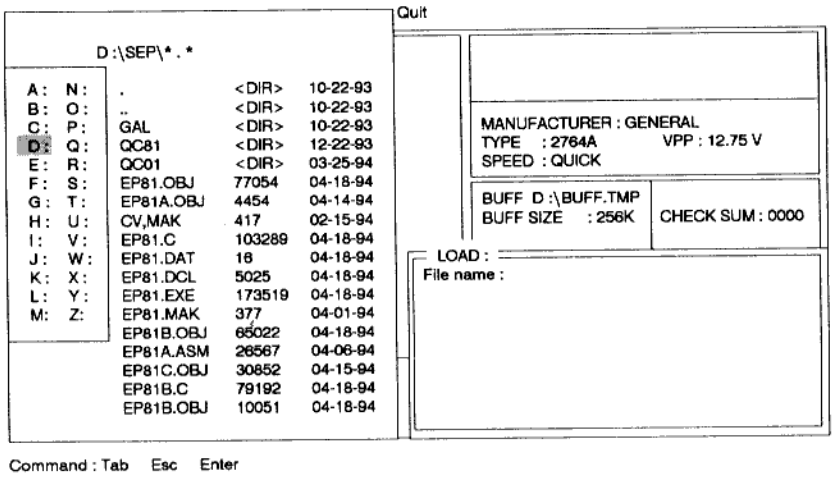


Subwindow of File

There are FOUR subfunctions under File for selection: Load file to buffer, Save buffer to file, list file Information, DOS.

4.1.1 Load file to buffer

To enter **Load file to buffer**, press **L** from the File subwindow or use the up/down arrow keys to move the highlighted bar to the desired function and then press <CR>. A dialog window for the File loading will appear on the screen.



Load file to buffer

NOTE

Files in following three formats can be loaded to the working buffer:

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 and use the following commands:

* <TAB> : switches control between the left and the right windows.

* <UP ARROW> & <DOWN ARROW> :

used to move the highlighted bar to a file or a subdirectory where a file is located.file is 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 NAME> : selects which drive's files will be shown in the selection window on the left.

NOTE:

These commands will only work when the TAB key is pressed and the left window is active/highlighted.

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 best to use this option. Files already in binary format load quicker to the programmer. All HEX files are actually converted to Binary by our software before being loaded to the buffer.

If 'B' is chosen you will be prompted "Load Adress(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 enter in the HEX number of the address where you would like the file to be loaded. The file will now be loaded to the buffer.

If 'I' is chosen you will be prompted for "File start seg.(0000):-, here the file start segment address is required. All data lying 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.(000000):-", here the actual start address of the file is required. All data lying between the start address that is given and the end of the file will be converted to binary and loaded to the buffer. The address range is from 0 to FFFFFF.

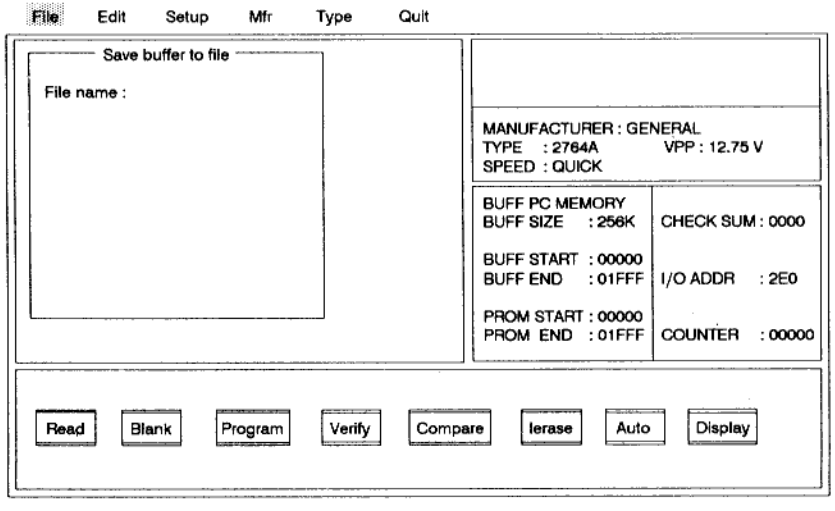
NOTE

1. In this function, file data is loaded byte by byte to the memory buffer without any converting. To convert a file from HEX to Binary ROM code file you must use HEXBIN.EXE or HEXBIN2.EXE.

2. Standard assemblers or compilers generate Intel HEX format files to be transmitted through In-Circuit Emulator or RS-232C. They can also generate Binary ROM code (details are available in your Assembler User's Manual), so you are recommended to use these tools.

4.1.2 Save buffer to file

To enter **Save buffer to file**, press **S** from the File subwindow or use the up/down arrow keys to move the highlighted bar to the desired function and then press <CR>. A dialog window for the Save file to buffer function will appear on the screen.



Save buffer to file

When prompted, enter the complete file name under which the file to be saved, including the drive letter and path names, if any, and then press <CR>. Enter the buffer start address of your desired portion, press <CR>, then enter its corresponding buffer end address and press <CR>.

When the screen displays the "OK!" message, the specified buffer contents have already been saved.

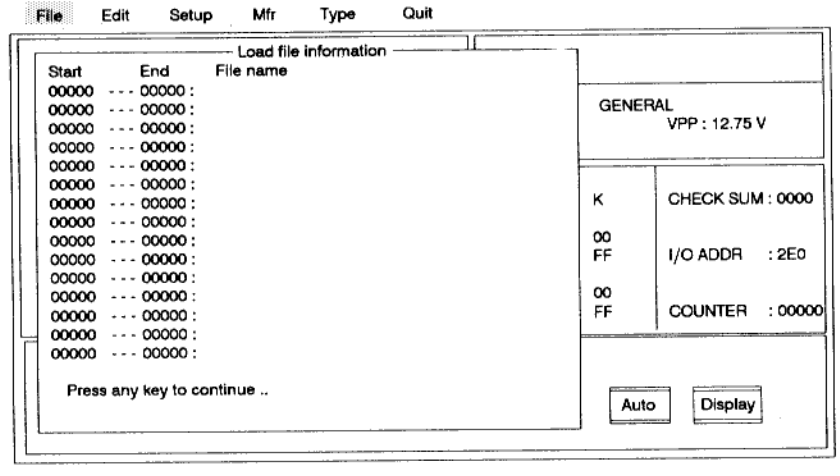
Press <ESC> to return to the main menu.

NOTE

Files saved under this function are exact Binary file without a convert.

4.1.3 list file Information

To enter **list file Information**, press **I** from the File subwindow, or use the up/down arrow keys to move the highlighted bar to the desired function and then press <CR>. The screen will list the latest 15 files that have been loaded.



list file Information

Press <ESC> to return to the main menu.

4.1.4 DOS

To enter **DOS shell**, press **D** from the File subwindow or use the up/down arrow keys to to move the highlighted bar to the desired function and then press <CR>.

After entering this function, the software will search for the file COMMAND.COM on the DOS boot disk drive. If it exists, the currently running main program will be transferred to the DOS environment. The screen will prompt as follows:

```
Type EXIT to return to the main menu

Microsoft (R) MS-DOS(R)  Version 3.30
        (C) Copyright Microsoft Corp. 1981-1987

A:\>
```

DOS shell

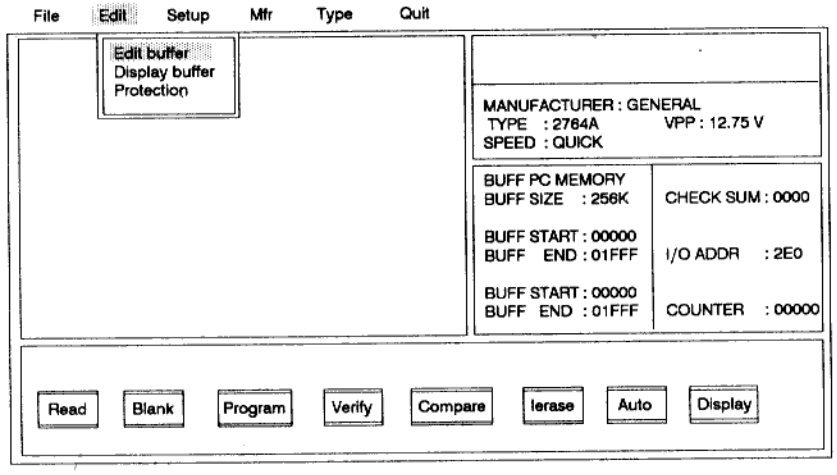
The main program is now controlled by DOS and waiting for your command. The command format is the same as that of the DOS command. Under the DOS command prompt, you can type EXIT and press <CR> to return to the main menu.

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.

4.2 Edit

To enter **Edit** function group, press **E** from the main menu or use the left/right arrow keys to move the highlighted bar to Edit and then press <CR>. The Edit submenu will appear on the screen.



Subwindow of Edit

4.2.1 Edit buffer

After entering the subfunction, select E again; The screen will display the Edit command summary.

```
                    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 cahracters
B   start, end, BINARY data         <RETURN>    : BINARY SEARCH MAX. 7 BYTES
.   filename [argu1] [argü2 ]...    <RETURN>    : SHELL
?                                   <RETURN>    : HELP
Q                                   <RETURN>    : QUIT
                    =======================================
* The information listed below is for reference only :
   The absolute start address of BUFFER : 2D8C : 0000
   The Buffer size : 256 K Bytes
= =
```

Edit command summary

You can proceed with the Edit procedure under command prompt = = by using the command format available above.

In addition to the command format, the actual buffer base address is also displayed at the bottom of the above screen. It is the real base address of the buffer in the PC memory and is displayed in SEGMENT:OFFSET form.

For special editing purposes, some experienced users need the address, particularly for use in their own editing program.

For example:
The user's editor is FRED.EXE, then he can use the DOT command to pass the base address:

FRED SEGMENT: OFFSET <CR>

The user may move this parameter from the command line to his own program in the same manner as executing his program under DOS command prompt:

A>FRED SEGMENT: OFFSET <CR>

Type Q and <CR> to return to the main menu.

4.2.2 Display buffer

To enter **Display buffer**, press **D** from the Edit submenu or use the up/down arrow keys to move the highlighted bar to the desired function and then press <CR>. Once you have entered the function, the buffer contents between the range from buffer start to buffer end address shown in the status field will be displayed on the screen, in binary format, as follows:

```
Press <ESC> to terminate display

00000  00 00 00 00 00 00 00 00  -- 00 00 00 00 00 00 00 00  ................
00010  00 00 00 00 00 00 00 00  -- 00 00 00 00 00 00 00 00  ................
00020  00 00 00 00 00 00 00 00  -- 00 00 00 00 00 00 00 00  ................
00030  00 00 00 00 00 00 00 00  -- 00 00 00 00 00 00 00 00  ................
00040  00 00 00 00 00 00 00 00  -- 00 00 00 00 00 00 00 00  ................
00050  00 00 00 00 00 00 00 00  -- 00 00 00 00 00 00 00 00  ................
00060  00 00 00 00 00 00 00 00  -- 00 00 00 00 00 00 00 00  ................
00070  00 00 00 00 00 00 00 00  -- 00 00 00 00 00 00 00 00  ................
00080  00 00 00 00 00 00 00 00  -- 00 00 00 00 00 00 00 00  ................
00090  00 00 00 00 00 00 00 00  -- 00 00 00 00 00 00 00 00  ................
000A0  00 00 00 00 00 00 00 00  -- 00 00 00 00 00 00 00 00  ................
000B0  00 00 00 00 00 00 00 00  -- 00 00 00 00 00 00 00 00  ................
000C0  00 00 00 00 00 00 00 00  -- 00 00 00 00 00 00 00 00  ................
000D0  00 00 00 00 00 00 00 00  -- 00 00 00 00 00 00 00 00  ................
000E0  00 00 00 00 00 00 00 00  -- 00 00 00 00 00 00 00 00  ................
000F0  00 00 00 00 00 00 00 00  -- 00 00 00 00 00 00 00 00  ................
00100  00 00 00 00 00 00 00 00  -- 00 00 00 00 00 00 00 00  ................
00110  00 00 00 00 00 00 00 00  -- 00 00 00 00 00 00 00 00  ................
00120  00 00 00 00 00 00 00 00  -- 00 00 00 00 00 00 00 00  ................
00130  00 00 00 00 00 00 00 00  -- 00 00 00 00 00 00 00 00  ................
00140  00 00 00 00 00 00 00 00  -- 00 00 00 00 00 00 00 00  ................
00150  00 00 00 00 00 00 00 00  -- 00 00 00 00 00 00 00 00  ................
Press any key to continue
```

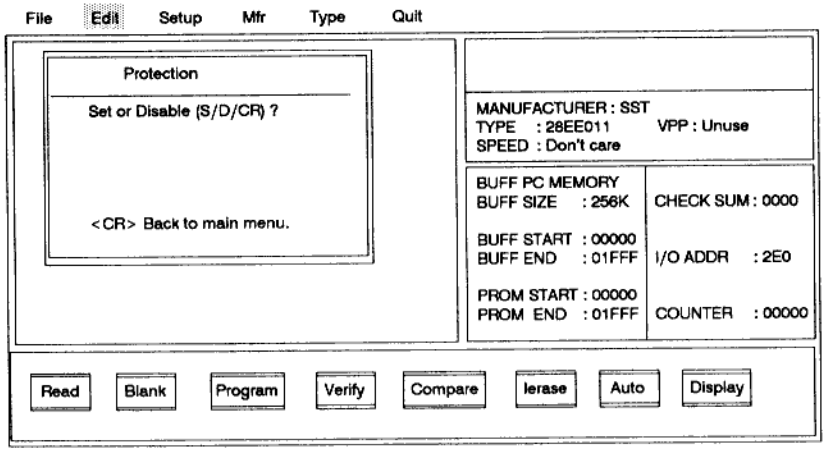
Display buffer

Press ^S to hold the display or press <ESC> to terminate the display. Then press any key to return to the main menu.

4.2.3 Protection

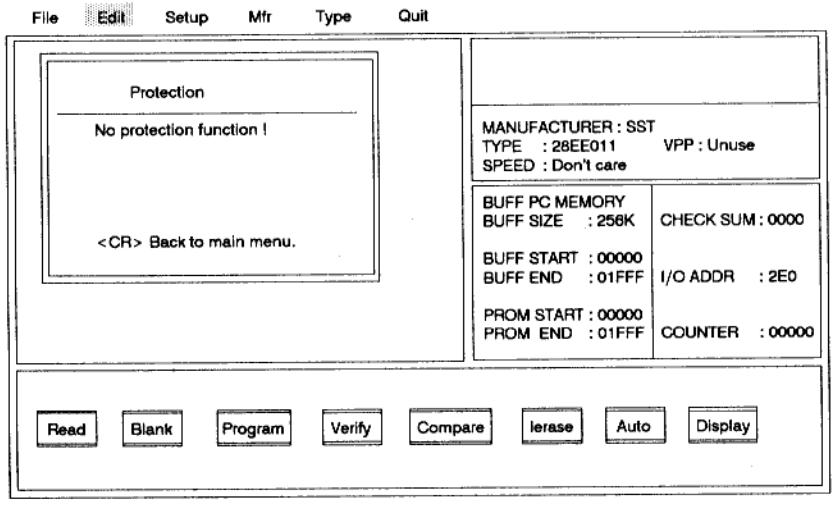
To enter **Protection**, press **P** from the Edit submenu or use the up/down arrow keys to move the highlighted bar to the desired function and then press <CR>. This function is to enable or disable the protection on EEPROM.

When the intended type has this function included, the screen will display as follow:



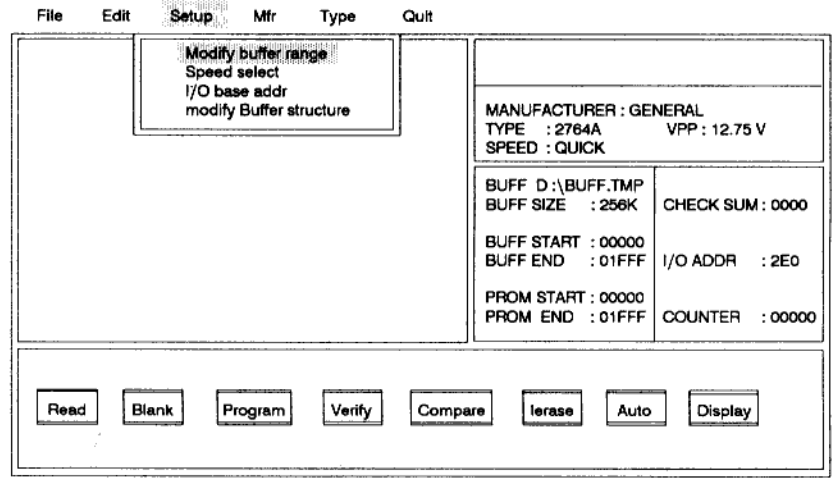
Press "S" to enable the protection or Press "D" to disable it, or press <CR> to return to the Edit menu.

If the intended type does not has the protection function, the screen will display as follow:



4.3 Setup

To enter the **Setup group function**, press **S** from the main menu or use the left/right arrow keys to move the highlighted bar to the desired function and then press <CR>. The Setup subwindow will appear on the screen.

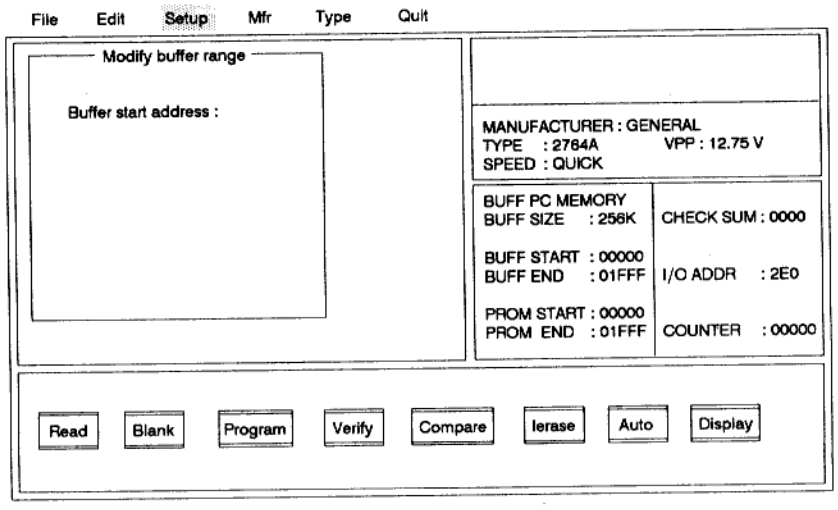


Subwindow of Setup

The four subfunctions under Setup are Modify buffer range, Speed select, I/O base address and modify buffer structure.

4.3.1 Modify buffer range

To enter **Modify buffer**, press **M** from the Setup subwindow or use the up/down keys to move the highlighted bar to the desired function and then press <CR>. A dialog window for Modify buffer will appear on the screen.



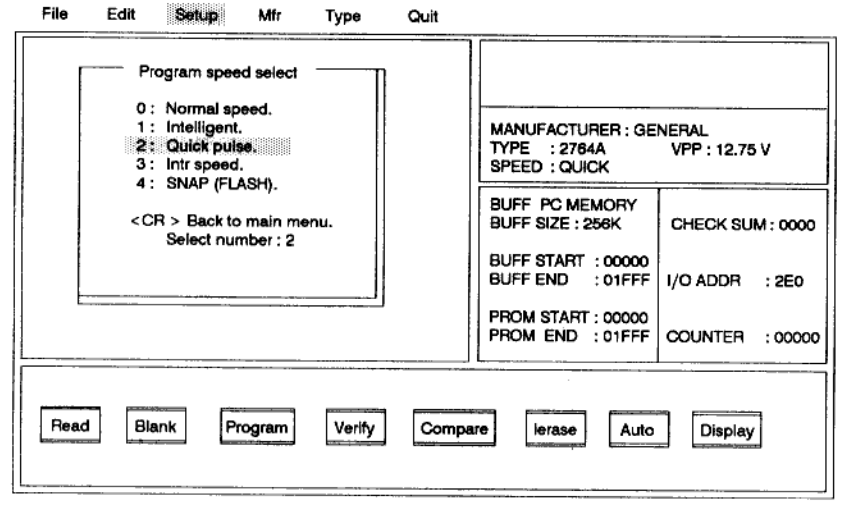
Modify buffer range

The prompted screen requires the user to enter the revised buffer start address of the EPROM and then press <CR>. All the values entered in HEX code will be updated in the status field for programming reference.

Press any key to return to the main menu.

4.3.2 Speed select

To enter **Speed select**, press S from the Setup subwindow or or use the up/down arrow keys to move the highlighted bar to the desired function and then press <CR>. The Speed select subwindow will appear on the screen.



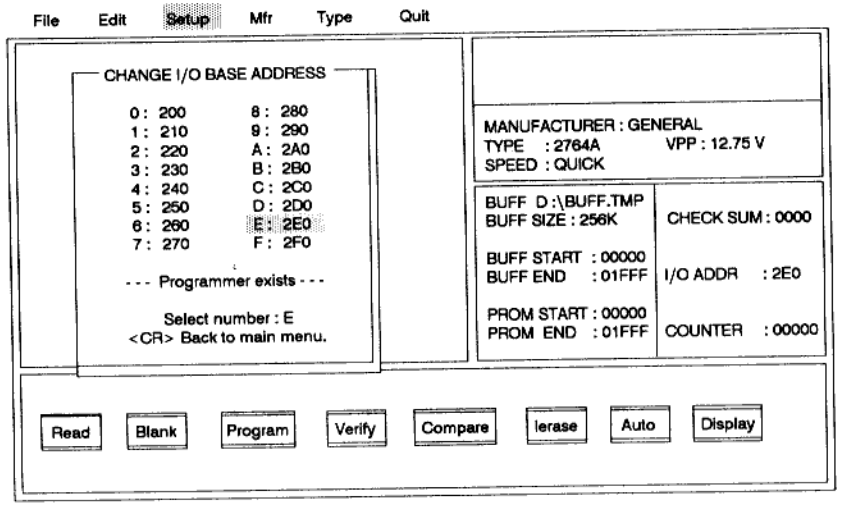
Speed select

NOTE

Most EPROMs have to apply the programming speed (Algorithm) as listed in the data sheet. Generally, they are grouped into Normal speed, Intelligent and quick pulse.

4.3.3 I/O base address

To enter **I/O base address** selection, press **I** from the Setup subwindow or use the up/down arrow keys to move the highlighted bar to the desired function and then press <CR>. The I/O base address subwindow listing all available I/O addressed will appear on the screen.



I/O base select

To select the I/O address set one of the DIP switches on the adaptor card in the on position. Only one position on the adapter card should be selected and turned on. The DIP switch positions correspond to I/O addresses as follows:

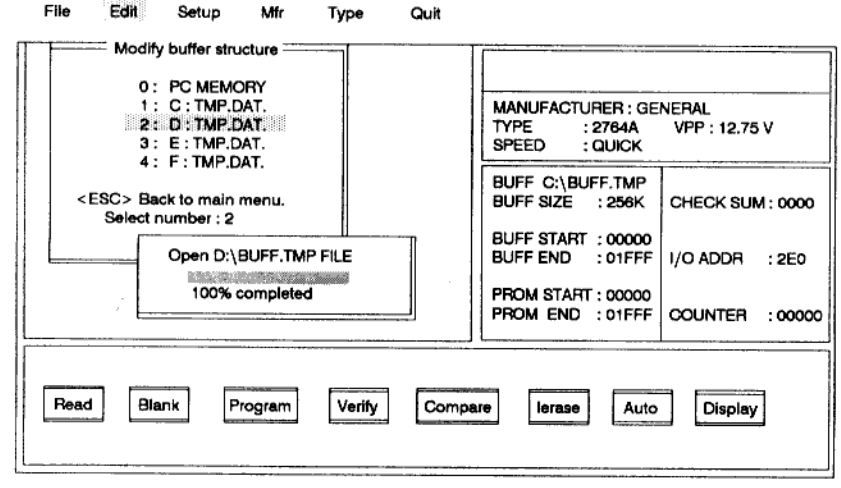
	DIP SW	POSITION	I/O ADDRESS
	SW1	1	200H
	SW1	2	210H
	SW1	3	220H
	SW1	4	230H
	SW1	5	240H
	SW1	6	250H
	SW1	7	260H
	SW1	8	270H
	SW2	1	280H
	SW2	2	290H
	SW2	3	2A0H
	SW2	4	2B0H
	SW2	5	2C0H
	SW2	6	2D0H
(DEFAULT)	SW2	7	2E0H
	SW2	8	2F0H

After entering the desired I/O address, it is necessary to execute the QUIT function which will save the revised I/O address in the parameter file EP8x.DAT for later use.

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

4.3.4 Modify Buffer Structure

This function allows the possibility to load files greater than 2 Mbit into the PC buffer.



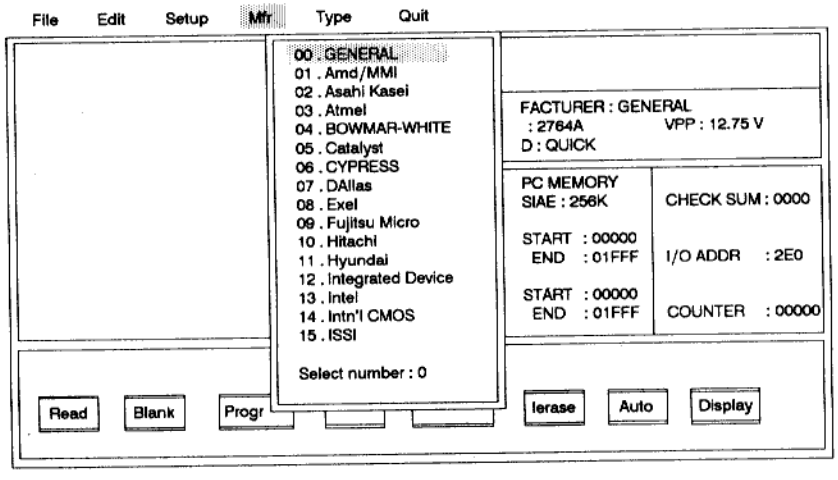
Subwindow of modify buffer structure

The modify buffer structure function is used to resolve this issue. First assign disk C:, D:, E:, F: as buffer, the buffer size increases with the disk capacity, so as to be able to handle EPROMs greater than 8 Mbit.

When the 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 be assigned as working buffer. After Quit function the parameter of the modify buffer structure will be saved to Buff.TMP. It will be automatically installed when the software is again used.

4.4 Mfr.

To enter **Mfr.** (Manufacturer), press **M** from the main menu or use the left/right arrow keys to move the highlighted bar to the desired function and then press <CR>. The Mfr. subwindow with currently supported manufacturers will appear on the screen.

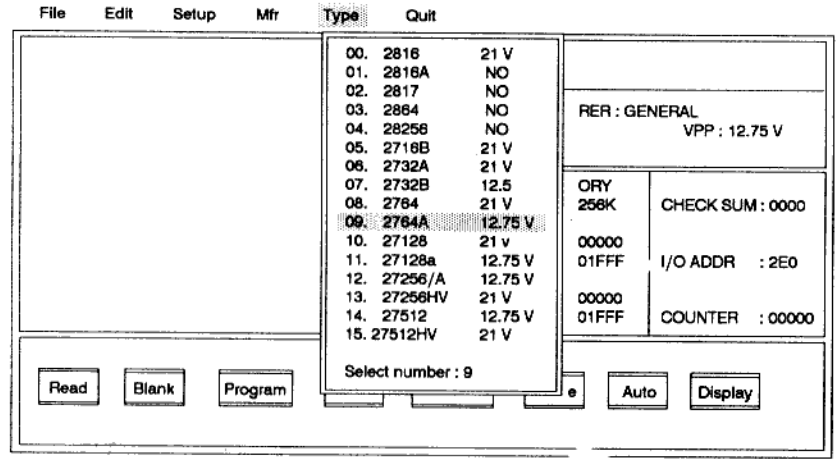


Subwindow of Manufacturer

Enter the number or letter which indicates a manufacturer corresponding to that of your EPROM. The selected manufacturer will then be updated in the status field for programming reference and a subwindow (as illustrated in Section 4.5) under function type will be displayed on the screen.

4.5 Type

To enter **Type**, press **T** from the main menu or use the left/right arrow keys to move the highlighted bar to Type and then press <CR>. The Type subwindow will appear on the screen.



Subwindow of Type

Select the number or letter which precedes the type corresponding to your EPROM type. The selected type and voltage will be automatically updated in the status field for programming reference.

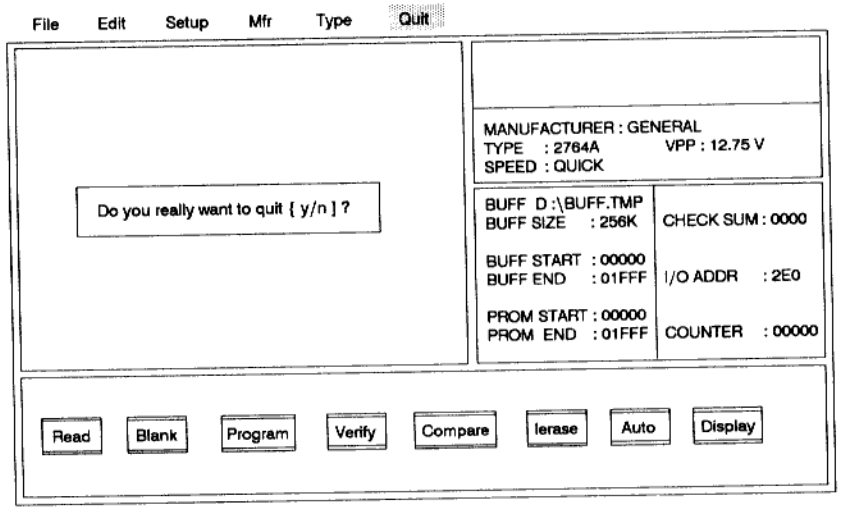
Press <ESC> or <CR> to return to the main menu.

NOTE

If the EPROM is not clearly labeled, it may be difficult to distinguish a EPROM 2732 requiring 25 V from a EPROM 2732A reguiring 21 V. In this case, the user should try to program the EPROM with the lower voltage first. If it does not work, then erase the EPROM and try the higher voltage.

4.6 QUIT

Press **Q** and enter **Y** from the main menu to exit the main menu and return to DOS. If you are in one of the subwindows, first exit that subwindow and then return to the main menu.

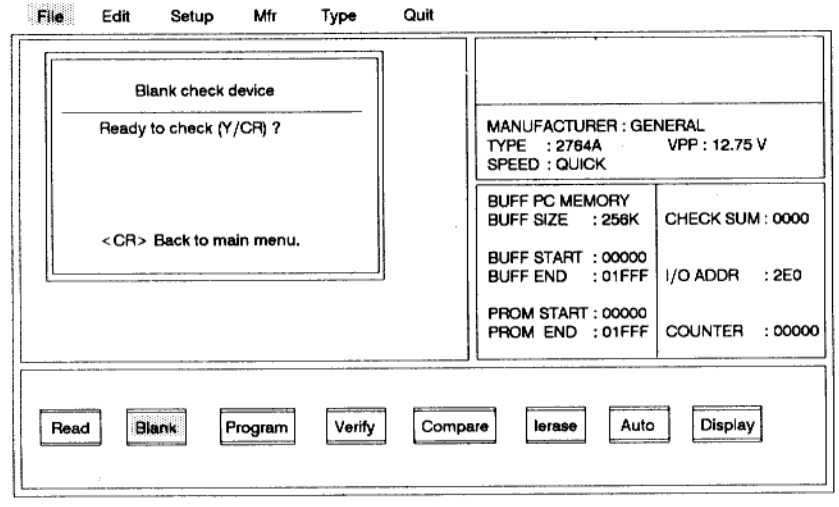


Subwindow of Quit

Before leaving for DOS, the main program will save all parameters including manufacturer, type, I/O base address and programming speed in the parameter file EP8x.DAT for later use.

4.7 Blank

To enter **Blank**, press **B** from the main menu. The Blank subwindow will appear on the screen.

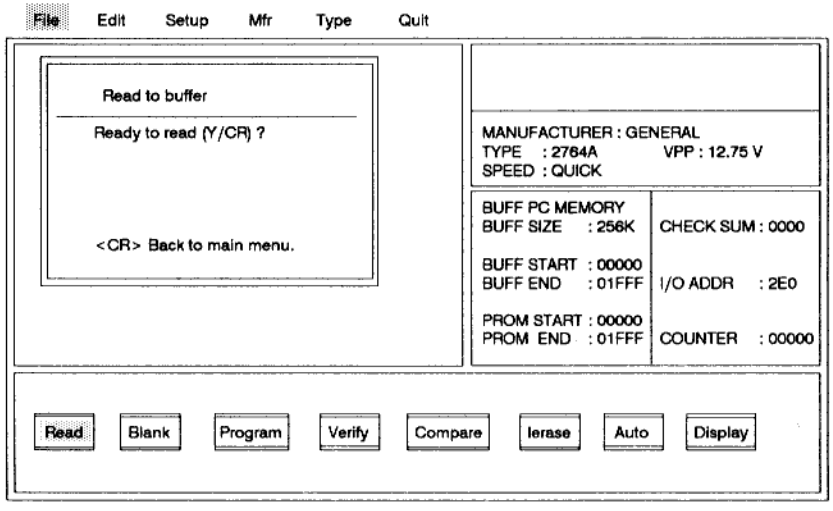


Subwindow of Blank

Press Y to start the blank check, or press <CR> to return to the main menu. If a chip fails the test, the first error address will be displayed, and the programmer module will continue to test the rest. If all pass, a "Blank check OK!" message will be displayed on the screen.

4.8 Read

To enter **Read**, press **R** from the main menu. The Read subwindow will appear on the screen.



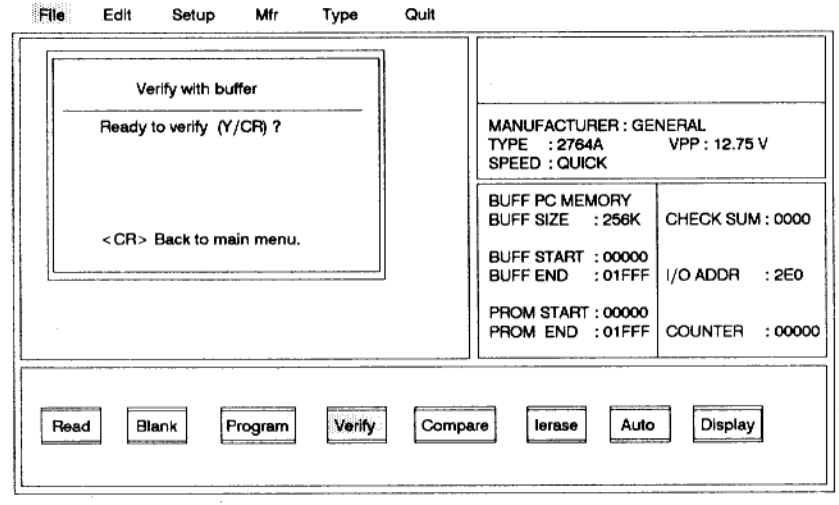
Subwindow of Read

Press Y to activate the function of reading data from the master EPROM to the buffer shown on the status field, or press <CR> to return to the main menu. The subwindow will show a "Reading now..." message during processing. After it has been completed, the "Read OK!" message will appear.

The check sum calculated during the processing will also be displayed in the status field after device reading.

4.9 Verify

To enter **Verify**, press **V** from the main menu. The Verify subwindow will appear on the screen.



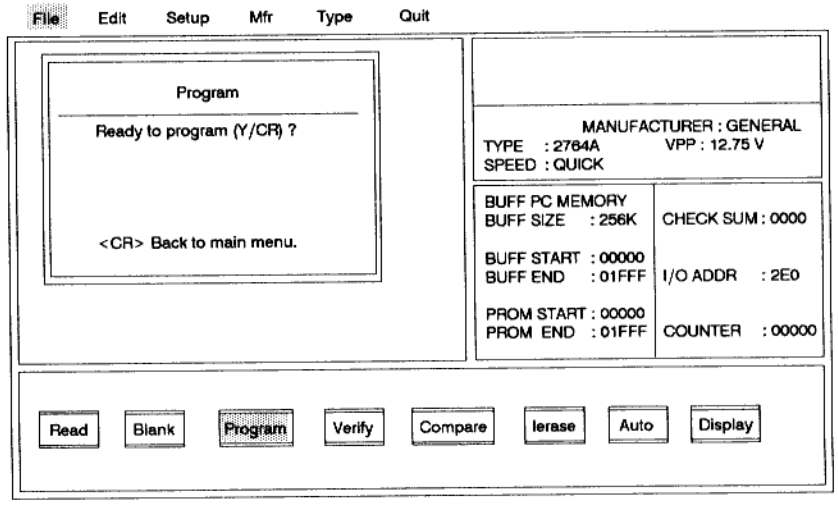
Subwindow of Verify

Press Y to verify the EPROMs with the buffer shown on the status field, or press <CR> to return to the main menu. During the operation, a "Verifying now" message will appear on the screen. Once it has been completed, the "Verify OK!" result will appear next to its associated socket number.

The Verifying routine will be terminated if the operation is error free.

4.10 Program

To enter **Program**, press **P** from the main menu. The Program subwindow will appear on the screen.

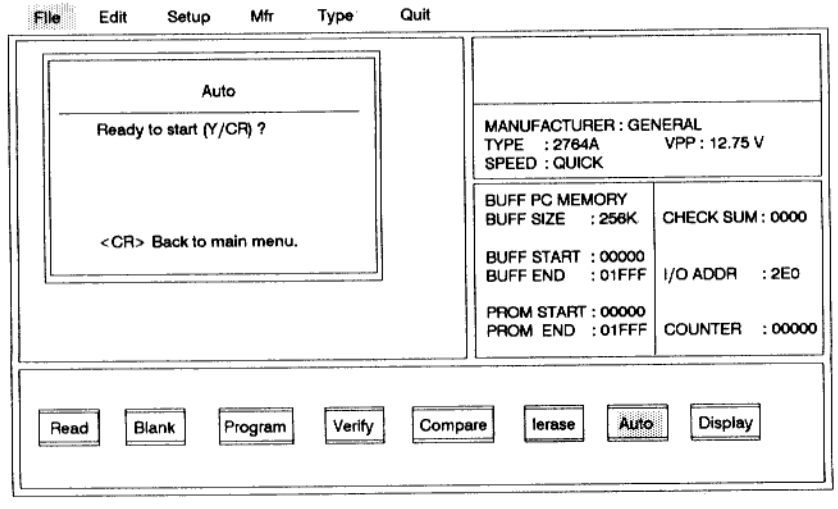


Subwindow of Program

Enter Y to activate the programming from the memory buffer to the EPROMs, or press <CR> to return to the main menu. A programming message as well as a scrolling counter number will simultaneously appear on the status field. If the transfer has been completed, a "Program OK!" message will appear next to its associated socket number.

4.11 Auto

To enter **Auto**, press **A** from the main menu. The Auto subwindow will appear on the screen.

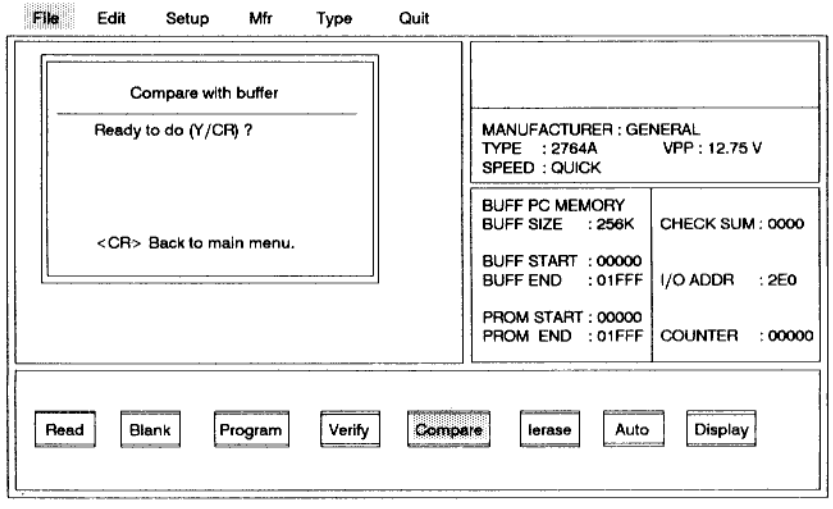


Subwindow of Auto

Press Y to activate the Auto function, or press <CR> to return to the main menu. This function is similar to the program function but it automatically blank checks before programming and verifies the data afterwards.

4.12 Compare

To enter **Compare**, press **C** from the main menu. The Compare subwindow will appear on the screen.



Subwindow of Compare

The programmer module will compare the buffer shown on the status field with the device in the master socket.

Press Y to compare the data in the device with that in the memory buffer, or press <CR> to return to the main menu. The screen will display the differences in the following format:

Error at:
Press <ESC> to terminate display

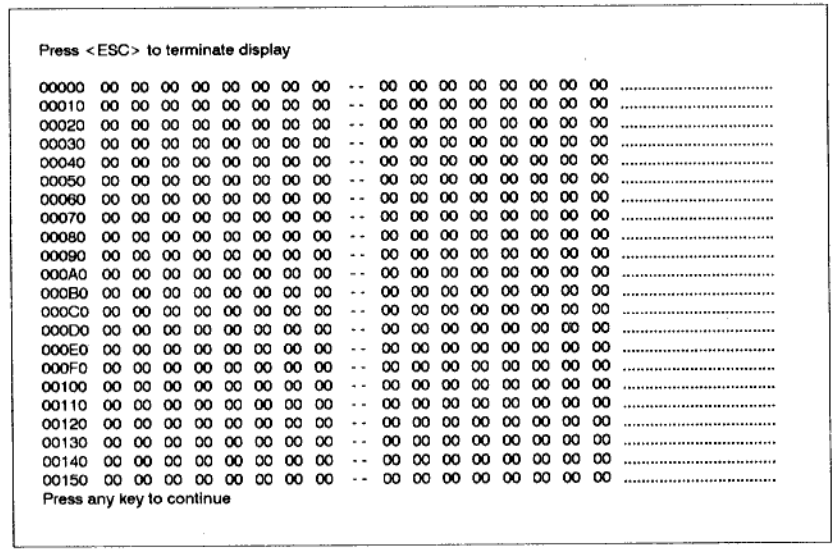
00000:DA - (00000:21) , 00001:1F - (00001:05)

Press any key to continue.

Press ^S to hold the display, or 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.

4.13 Display

To enter **Display**, press **D** from the main menu. The Display subwindow will appear on the screen.

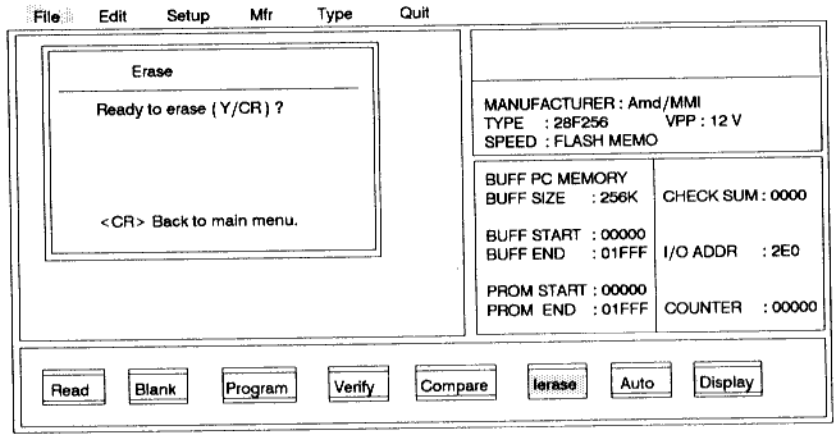


Subwindow of Display

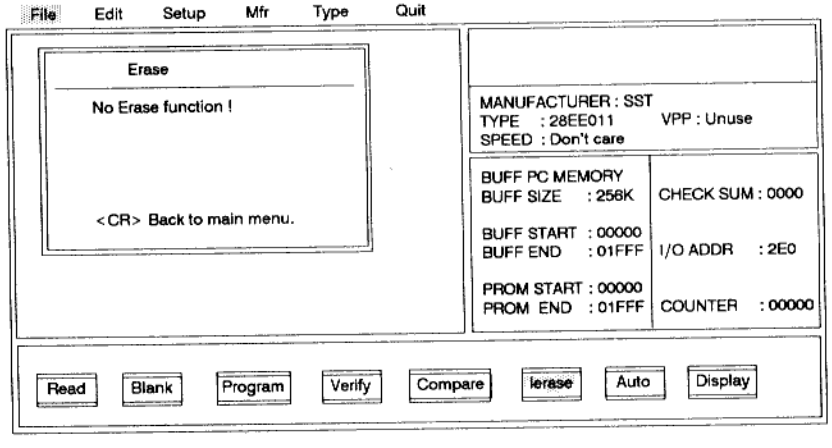
The screen displays the address and current contents of the master EPROM. Press <ESC> to exit the display and press any key to return to the main menu.

4.14 Erase

To enter **Erase**, press **I** from the main menu. This function is to enable or disable the erase function on EEPROM. The screen will display as follow:



Press <CR> to return to the Edit menu.
If the intended type does not has the erase function, the screen will display as follow:



APPENDICES

APPENDIX A. UTILITY PROGRAM

A.1 HEX TO BINARY code converters

Before any information is loaded to the working buffer it is first converted to binary format. These HEX converters will convert your HEX format files to executable BIN code that is recognizable to your target CPU. Even though our 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 they will be downloaded at a faster rate to the working buffer. If, for example, a user has a 2Mbit or larger file that needs to be split into several files these HEX to BIN converters will be useful to produce files that our other Split and Shuffle utilities can work on. (SPLITX.EXE and SHUFFX.EXE only accept binary files as input.)

Two HEX file converters are included with your software. The first is HEXBIN.EXE which has a maximum conversion size of 64K bytes. This converter can convert the following file formats to binary(BIN) format files:

INTEL HEX format
MOTOROLA S HEX Formats
TEKTRONIX HEX format

HEXBIN2.EXE can convert INTEL, MOTOROLA, and TEKTRONIX extended HEX format files to binary files with a maximum conversion size of 256 Kbyte. Another difference between HEXBIN2.EXE and HEXBIN.EXE is that HEXBIN2.EXE allows you to specify a starting address, the leading "garbage" of the file will be skipped to maintain a small sized output binary file.

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] <CR>

All entries in brackets are options.

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 resulting output binary file.

HEX format is: I for INTEL HEX
M for MOTOROLA S HEX(S records)
T for TEKTRONIX HEX

Start address and end address are Hexadecimal digit.

Note: In the INTEL extended HEX format, the start address represents the start segment address. All data lying between the start segment X 16 and the end of the file will be converted to binary format.

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

For example:

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

HEXBIN2 will convert the HEX file DEMO.HEX to the binary file DEMO.BIN using the INTEL HEX converting technique, and only data lying after 10000H 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 FORMAT (<I>NTEL <M>OTOROLA <T>EKTRONIX) : I

A.2 Dump Binary File To Console

Executable BINARY files can not be displayed on the screen using a DOS TYPE

command. DUMP.EXE can convert a BINARY file to HEXADECIMAL characters that can be displayed on the screen or by a printer.

The input command under the DOS command prompt is:

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

All entries in brackets [ ] are options.

^P : <CTRL> + P <CR> : return or Enter key

The ^P option can be used to connect the PC with the printer, the data displayed on the screen will also be output to the printer.

FILENAME is a standard filename as specified in DOS.

Start address is a hexadecimal digit that specifies what address the display will start at.

A.3 2-way, 4-way Binary File Splitter

SPLIT2.EXE can split a 16-bit binary source file into two 8-bit binary files. One file is the collection of data lying on the LOW byte of the 16-bit words in the source file. The other file is the collection of data lying on the HIGH byte of the 16 bit words in the source file. The utility is generally used to split a 16 bit file into two 8 bit files that can be programmed to an EVEN EPROM and an ODD EPROM.

SPLIT4.EXE can split 32-bit binary format source files into four 8-bit binary files. The first file is the collection of data lying on the 1st byte of the 32-bit words in the source file. The second file is the collection of data lying on the 2nd byte of the 32-bit words in the source file. The third file is the collection of data lying on the 3rd byte of a 32-bit words in the source file. The fourth file is the collection of data lying on the 4th byte of a 32-bit word in the source file.

The input command under the DOS command prompt is:

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

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

A.4 2-way, 4-way Binary File Shuffler

SHUFF2.EXE can shuffle two 8 bit binary source files into a 16-bit binary file. The first 8-bit file will be collected to the LOW bytes of the 16 bit words in the output file. The second 8-bit file will be collected to the HIGH bytes of the 16-bit words in the output file.

SHUFF4.EXE can shuffle four 8-bit binary format source files into one 32-bit binary file. The first 8-bit file will be collected to the 1st byte of the 32 bit output file. The second 8-bit file will be collected to the 2nd byte of the 32 bit output file. The third file will be collected to the 3rd byte of the 32 bit output file. The fourth file will be collected to 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.

APPENDIX B. LIST OF SUPPORTED DEVICES

GENERAL

2816	21 V	2732B	12.5V	27512	12.75V
2816A	NO	2764	21 V	27512HV	21 V
2817	NO	2764A	12.75V	27C010	12.75V
2864	NO	27128	21 V	27C020	12.75V
28256	NO	27128A	12.75V	27C040	12.75V
2716B	21 V	2756/A	12.75V	27C080	12.75V
2732A	21 V	27256HV	21 V		

AMD/MMI

2716B	12.5V	27C256	12.75V	2864A	NO
2732A	21 V	27512	12.75V	2864B	NO
2732B	12.5V	27C512	12.75V	2864AE	NO
9732A	21 V	27LV512	12.75V	2864BE	NO
9732B	12.5V	27C513	12.75V	28F256	12 V
2764	21 V	27C010	12.75V	28F512	12 V
9764	21 V	27LV010	12.75V	28F010	12 V
2764A	12.75V	27C020	12.75V	28F020	12 V
27128	21 V	27C040	12.75V	29F010	NO
27C128	12.75V	27C080	12.75V	29F040	NO
27128A	12.75V	2817A	NO		

ATMEL

27HC64	13.0V	27C020	13.0V	28MC010	NO
27C128	13.0V	27C040	13.0V	28C010	NO
27256	13.0V	27LV040	13.0V	28MC020	NO
27C256	13.0V	27C080	13.0V	28MC040	NO
27LV256	13.0V	28C04	NO	29C256	NO
27HC256	13.0V	28C16	NO	29C257	NO
27C512	13.0V	28HC16	NO	29LV512	NO
27C512R	13.0V	28C17	NO	29C512	NO
27LV512R	13.0V	28HC17	NO	29C010	NO
27C513	13.0V	28C64	NO	329C040	NO
27C513R	13.0V	28HC64	NO	28LV256	NO
27C010/L	13.0V	28C256	NO		
27LV010	13.0V	28HC256	NO		

Asahi Kasei

28C64	NO

BOWMAR-WHITE

8023	NO

CATALYST

2764A	12.75V	27512	12.75V	28C17A	NO
27128A	12.75V	27010	12.75V	28C64A	NO
27256	12.75V	27HC010	12.75V	28C256	NO
27HC256	12.75V	28C16A	NO	28F010	12 V

CYPRESS

CY7C256	12.5V	CY27C128	12.5V	CY27H010	12.5V

DALLAS

DS1220AB/AD	NO	DS1225D/E	NO	DS1230Y/AB	NO

EXEL

28C04	NO	2816A	NO	2865A	NO
2804A	NO	2817A	NO		
28C16	NO	2864A	NO		

FUJITSU

2732A	21 V	27256	12.5V	27C1001	12.5V
27C32A	21 V	27C256	21 V	27C4001	12.5V
2764	21 V	27C256A	12.5V	2864A	NO
27C64	21 V	27C256H	12.5V	28C64	NO
27128	21 V	27C512	12.5V	2865A	NO
27C128	21 V	27C1000	12.5V	28C65	NO

HITACHI

482732A	21 V	27C256	12.5V	29C101	12 V
27C64	21 V	27512	12.5V	29C101B	12 V
482764	21 V	27C101	12.5V	27C4001	12.5V
27128A	12.5V	27C101A	12.5V	28F101	12V
4827128	21 V	27C301	12.5V	28F4001	12V
27256	12.5V	27C301A	12.5V		

Hyundai

27C64A	12.75V

Integrated Device

78C16A	NO

Intel

2816A	NO	27128	21 V	27C100	12.75V
2817A	NO	27128A	12.75V	27C010	12.75V
2732A	21 V	27128B	12.75V	27C010A	12 V
2732B	12.5V	27C128	12.75V	27C020	12.75V
2764	21 V	P27128A	12.75V	27C040	12.75V
2764A	12.75V	27256	12.75V	27C080	12.75V
27C64	12.75V	27C256	12.75V	28F256	12 V
87C64	12.75V	P27256	12.75V	28F512	12 V
87C256	12.75V	27512	12.75V	28F010	12 V
87C257	12.75V	27513	12.75V	28F020	12 V
68C257	12.75V	27010	12.75V	28F001BX-B	12V
P2764A	12.75V	27011	12.75V	28F001BX-T	12V

Intn'l COMS

27CX256	12.75V	27CX010	12.75V

ISSI

27HC010	12.75V

Itaucom

27512	12.75V

MATSUSHITA

2764	21 V	27128	21 V

Microchip

27C64	13.0V	27LV256	13.0V	28HC16	NO
27HC64	13.0V	27C512	13.0V	28C17/A	NO
27LV64	13.0V	27C512A	13.0V	28HC17	NO
27C128	13.0V	27LV512	13.0V	28CP64	NO
27256	13.0V	27C513	13.0V	28C64/A	NO
27C256	13.0V	28C04/A	NO	28C256	NO
27HC256	13.0V	28C16/A	NO	28CP256	NO

MITSUBISHI

2732A	21 V	27C256	12.5V	27C201	12.5V
2764	21 V	27512	12.5V	27C401	12.5V
27128	21 V	27C512A	12.5V	28C64A	NO
27C128	21 V	27C100	12.5V	28F101	12 V
27256	12.5V	27C101	12.5V		

Motorola

TMS2716	12.5V

MXIC

MX27C256	12.75V	MX27L1000	12.75V	28F1000	12 V
MX27L256DC	12.75V	MX27C1001	12.75V	27C8000	12.75V
MX27C512A	12.75V	MX27C2000	12.75V	28F1000	12V
MX27L512DC	12.75V	MX27L4000DC	12.75V	28F4000	12V
MX27C1000	12.75V	MX27C4000	12.75V		

National

2816	NO	MC27C128C	13 V	NM27LC512	12.5V
2864	NO	NM27C128	13 V	NM27LV512	12.5V
NMC9816A	NO	NMC27CP128	12.5V	NMC27C1023	12.75V
NMC9817	NO	NM/NMC27C256	13 V	NMC27C010	12.75V
NMC9817A	NO	NMC27C256B	12.5V	NM27C010	12.75V
NMC98C64	NO	NM27LC256	12.5V	NM27LV010	12.75V
NMC27C16B	13 V	NMC27CP256	12.5V	NMC27C020	12.75V
NMC27C32B	13 V	NM/NMC27C512	13 V	NM27C040	12.75V
NMC27C64	13 V	NM/NMC27P512	13 V	NM27P040	12.75V
NMC27CP64	12.5V	NMC27C512A	12.5V		

NEC

2732A	21 V	27C512	12.75V	27C8001	12.75V
2764	21 V	27C1000/A	12.75V	28C04	NO
27128	21 V	27C1001/A	12.75V	28C05	NO
27256	21 V	27C2001	12.75V	28C64	NO
27256A	12.75V	27C4001	12.75V	28C256	NO

OKI

2816A	NO	27256	12.75V	271000	12.75V
28C16A	NO	27C256H	12.75V	271000ZB	12.75V
28C64A	NO	27C256HZB	12.75V	27C101	12.75V
28C256	NO	27C256ZB	12.75V	2127C101ZB	12.75V
2732A	21 V	27256ZB	12.75V	27101H	12.75V
2764	21 V	27256ZC	12.75V	27C2000	12.75V
2764A	12.75V	27512	12.75V	24.27C2000ZB	12.75V
27128	21 V	27512ZB	12.75V	27C401	12.75V
27128A	12.75V	27C1000	12.75V	27C401ZB	12.75V

OMNI WAVE SEMICONDUCTOR

27C256	12.75V	27C101	12.75V

PANASONIC

27C64A	12.75V	27C256	12.75V	27C512	12.75V

Richo

27C32	21 V	27C64	21 V	27C256	12.75V

ROCKWELL

2816A	NO

SAMSUNG

28C16	NO	28C64	NO	2865A	NO
2816A	NO	2864A	NO	2865AH	NO
28C17	NO	2864AH	NO	28C256	NO
2817A	NO	28C65	NO	29C010	NO

SEEQ

2764	21 V	5517A(H)	NO	38C16	NO
27128	21 V	2864	NO	38C32	NO
27C256	12.75V	2864H	NO	52B13	NO
28C04	NO	28C64	NO	52B33	NO
2816A(H)	NO	28C65	NO	28C010	NO
5516AH	NO	28256	NO		
2717A(H)	NO	28C256	NO		

SEIKO

S2840R/A	NO	S28F512A	12 V

SGS

2732A	21 V	27C256	12.75V	28F256	12V
2764	21 V	27512	12.75V	28F256A	12V
2764A	12.75V	27C1000	12.75V	28F512	12V
27C64	12.75V	27C1001	12.75V	28F101	12V
27128A	12.75V	27C2001	12.75V	28F1001	12V
27256	12.75V	27C4001	12.75V	28F201	12V

SHARP

5762	12.75V	57126	12.75V	57254	12.75V
5763	12.75V	57127	12.75V	57255	12.75V
5764	12.75V	57128	12.75V	57256	12.75V

PHILIPS

27C64	21 V	27C256	12.75V	27C010	12.75V
27C64A	12.75V	27C256F	12.75V		
27C64AF	12.75V	27C512	12.75V		

S-MOS

27C64H	21 V	27128H	21 V	27C256H	12.75V

Sony

CXK27C256DQ	12.75V	CXK27C1000DQ	12.75V
CXK27512DQ	12.75V	CXK27C1001DQ	12.75V

SST

28EE011	NO

TI

27(P)32A	21 V	27C256	13.0V	27PC040	13.0V
27(P)64	21 V	27C512	13.0V	28F010	12 V
27C64	13.0V	27PC512	13.0V	29C256	NO
27128	21 V	27C010A	13.0V	29C257	NO
27C128	13.0V	27PC010A	13.0V	29C010	NO
27PC128	13.0V	27C020	13.0V		
27256	13.0V	27C040	13.0V		

TOSHIBA

2732A	21 V	27256A	12.75V	532000	13.0V
2464	21 V	27256B	12.75V	534000	12.75V
2464A	12.5V	27256(D)	21 V	541000	12.75V
2764A	12.75V	54256	21 V	541001	12.75V
2764D	21 V	54256A	12.75V	571000	12.75V
24128	21 V	57256	21 V	571001	12.75V
24128A	12.75V	57256A	12.75V	574000	12.75V
27128A	12.75V	24512	12.75V	544000	12.75V
27128D	21 V	27512	12.75V	578000	12.75V
24256	12.75V	27512A	12.75V		
24256A	12.75V	57512A	12.75V		

VLSI

27C64	12.75V	27C256	12.75V	28H64	NO
27C128	12.75V	27C512	12.75V		

WSI

57C128	12.75V	27C512	12.75V	27C010L	12.75V
27C256	12.75V	27C512F	12.75V	27C010F	12.75V

Winbond

27F256	12V	29EE011	NO	27F512	12V
27F010	12V				

XICOR

2816A	NO	2864B	NO	28C256	NO
2816B	NO	2864H	NO	28C010	NO
2864A	NO	28256	NO		

APPENDIX C. TROUBLE SHOOTING

We have provided the following troubleshooting guide to help you overcome some of the more commonly experienced 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

PROBLEM 1 :
When I turn my computer on, I get 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 your chip is correctly installed and the handle is down.

1-3 Your power supply may not have sufficient power to drive both your system and the 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 base address set correctly for your programmer. Double check the I/O base address assignment.

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 There may not be good 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 slowing your system down as much as possible, or try using 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 speeds greater than 486 33 MHz.

PROBLEM 3 :
When I install the system adapter card, some of my other peripherals start behaving strangely!

RESOLUTION 3 :
3-1 You are probably experiencing an I/O base address conflict. Double check the I/O base address 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, 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 system adapter card fully seated in its slot?
2. Are all cable connections securely fastened?
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 have the same I/O base address as the system adapter card?

APPENDIX D. GLOSSARY

The most commonly used terminologies are explained here so that the user may refer to them whenever necessary.

*** Memory Buffer**
The buffer is a block area of PC memory allocated by the main program through DOS. This buffer is used by the main program as an intermediate storage.

The main program can read the DEVICE contents to the buffer and save it onto 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 have the advantage of manipulating the buffer contents at will. The modified buffer can always be saved to a disk file for future reference.

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

Since the memory 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 memory buffer.

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

*** Device Start and Device End Addresses**
The start and end addresses are offset addresses specified for the DEVICE contents.

*** Check Sum**
This value 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.

*** I/O Address**
This is the I/O base address of the system adapter 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 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 increasingly displayed on the 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 APPENDIX B for the details for each EPROM supported.