

USER'S MANUAL

UNIVERSAL PROGRAMMER & IC TESTER
P – 12S

1. INTRODUCTION

This manual provides instructions for installing and operating the programmer on an **IBM PC** or compatible running Windows 95, 98 or Windows NT. Communication with the host PC is through **RS 232C** data transmission (baud rate up to 115, 200 bps). Inside the programmer, there is a high-speed processor with a **1Mbit memory** buffer. All timing and programming control is done in the programmer to ensure accurate and precise programming waveforms. The 1Mbit buffer supports programming of memory devices and controllers with up to 1Mbit memory as well as most PLD devices. The software will automatically use the PC memory as the buffer for devices having more than 1Mbits. The programmer's pin driver circuits are fully programmable to support standard 5V IC's as well as low voltage IC's.

2. INSTALLATION

The programmer is easy to install and use with any IBM 486/Pentium computer or compatible running Windows 95, 98 or Windows NT. You are familiar with the installation of PC add-ons and software in PCs that run the windows operating systems.

2.1 Host System Requirements

- IBM 486/Pentium or compatible PC (recommended Pentium or more powerful PC)
- Microsoft Windows compatible mouse is recommended.
- Approximately 10 to 20 Mbytes free disk space.
- Operating system: Microsoft Windows 95, 98 or Windows NT.
- 1 Serial port (RS232).
- 64 MB DRAM memory or above is recommended.

2.2 Hardware Installation Procedures

Step 1:

With the host PC and programmer turned off connect the male connector of the DB-9 cable to the female connector of the DB-9 connector to the programmer. Connect the female connector of DB-9 cable to RS232C port of your host PC.

Step 2:

First, power on the computer, then the programmer. The power supply of the programmer can accept 185VAC – 265VAC, 47Hz – 65Hz Power sources.

Step 3:

Check LEDs on the programmer.

- POWER LED must be ON.
- BUSY LED must be OFF.
- DONE LED is in OFF state

If the LEDs are not in the correct state, turn off the PC and the programmer and check all connections between the serial port and the programmer. If an additional RS-232C card is being used then check that card is securely installed in the PCI bus slot in the PC. Then turn ON the computer and the programmer to check the LEDs on the programmer again. If the LEDs are still not in the proper state, you may have a conflict with another peripheral on the PC set to the same COM port address as the port you are using.

2.3 Software Install Procedure

Insert programmer install CD to CD Drive or copy all the files to Hard Disk

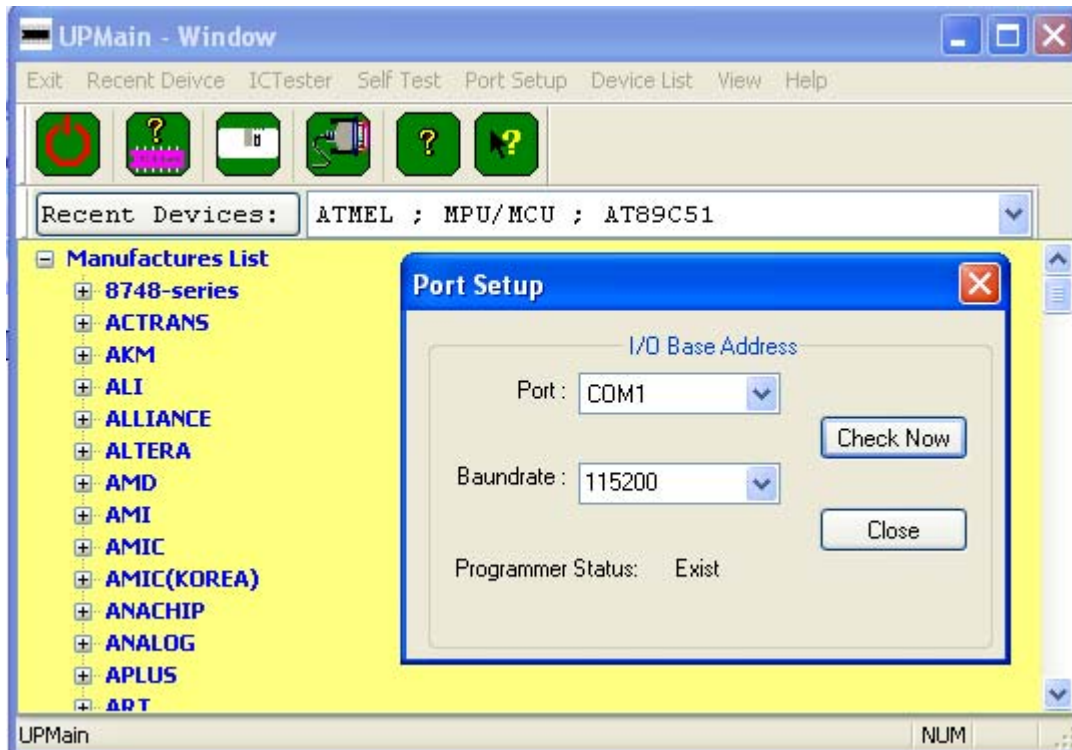
Run “UP.EXE”, The “SETUP” program will open a Window; follow the install instruction step by step to install your programmer software into your computer. For WINDOWS NT to install the software, you must be logged on as an administrator.

2.4 Hardware identification from Main menu.

Run "UNIVERSAL PROGRAMMER & IC TESTER" software, the main window will appear as below.



Click on the "Port Setup" menu and select the "0. I/O Base Address"



The current I/O base address registered in the software will appear on the screen and software will automatically check hardware identification circuits in the programmer. The software will display Programmer Status: "Exist" or "Programmer Status: Not Exist". When "Exists" appear in the status window that means the hardware installation is OK. The "Not Exist" message appears when the hardware I/O address (COM port address) does not match the I/O address selected in software, when the RS-232C hardware COM port or IRQ setting conflicts with other add-on cards inserted in the PC, if the cables and interface connections are not firmly fastened, if the programmer is not turned on.

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

1. Make sure the programmer is well connected to the programmer docking system, the power is turned on and the green ON LED is ON.
2. Check if the RS-232C connection between programmer and PC is properly connected.
3. If no COM port settings work, please try setting the baud rate to a slower speed. Some PC's serial port hardware cannot support 115K baud transmission speeds.
4. Turn the programmer power off and then on again. When the software is exited the COM port and baud rate setting will be saved as the default settings for future use. * Refer to the Function Reference Guide for more details about the change I/O address function.

3. USING THE PROGRAMMER

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

3.1 Definition of Key Symbols and Terms Used for Programming

3.1.1 Device Symbols for EPROM, EEPROM, BPROM and MPU

DEVICE: An integrated circuit (IC) that will be programmed or copied.

Buffer: Buffer A block of memory inside the programmer controller allocated by the device driver file. The buffer is used by the device driver file as an intermediate storage. When you want to program a data file to a DEVICE you first load the file to the buffer. The contents in the buffer are then programmed to the target DEVICE. When you read the contents of a master DEVICE the data is stored in the buffer. The buffer can then be edited or saved to disk for future reference.

Buffer Start and Buffer End Addresses: The buffer start and end addresses are offset addresses specified from the base address of the Buffer. The data stored between the buffer start and buffer end addresses contains the subset of the programmer memory buffer that is programmed to a target device. This is also the area of the buffer included in checksum calculations.

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

Bit count of the data contents of several DEVICES:

NIBBLE WIDE PROM is 4.

BYTE WIDE PROM is 8.

WORD WIDE PROM is 16.

MPU is usually 8 bits but can be 12, 14, or 16 or some other count depending on the device.

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

NOTE: In order to get correct programming data, some device files forbid users to change the Buffer Start, Buffer End, Device Start, and Device End settings.

EVEN and ODD address mapping: The address sequence of data contents can be assigned to CONTINUOUS (NORMAL), EVEN or ODD (note: not all devices have EVEN/ODD functions) whenever you want to READ, PROGRAM or VERIFY the EPROM. For example, in a program function selection the software will give you three selections --- Normal, Even, or Odd.

You may click on Normal (Continuous) to program the data in the buffer to the device as follows:

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

You may click on Even to program the data in the EVEN addresses to the device as follows:

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

You may click on Odd to program the data in the ODD addresses to the device as follows:

Buffer address	+ 1	is programmed to	Device start	+ 0
	+ 3			+ 1
	+ 5			+ 2

Troubleshooting Hint:

Whenever using the even or odd functions, the software will automatically expand your working buffer to twice as large as your target device address space. For example, if your target device is a 27C256 with addresses from 0h to 7FFFh you want the buffer start address to be 0000h and the buffer end address to be FFFFh which is the address space of a 27C512.

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

Counter: This is the programming address counter. During DEVICE programming, the counter on the screen will scroll to show the current address in the target device that is being programmed. (Note: not all software has this function)

MFR & TYPE: Every DEVICE has its own manufacturer (MFR) and type number (TYPE). Please refer to Chapter 4 for detailed descriptions.

Security Fuses: Security fuses are present in many microcontroller devices. When a security fuse is programmed the data stored inside the microcontroller can not be read by a device programmer for reverse engineering purposes. Generally a device will read as blank if the security fuse is blown. The device will operate functionally equivalent with the security fuse intact or blown.

Some UV erasable devices can not be programmed again once the security fuse is programmed. Please check with your data book before programming the security fuse of a UV erasable (windowed) microcontroller.

Lock Bits: Some microcontrollers use lock bits in place of a single security fuse. Usually you have the option of selecting to program 1 lock bit or all lock bits to provide different levels of code protection. The definition of these bits varies from manufacturer to manufacturer. Please consult the data book of your target device for more details.

For some devices like Microchip PIC series devices, if your Lock Bits in the data file are set, the checksum will then change because of the PIC devices ' checksum counting rules.

Encryption: Encryption is another form of code protection found in some microcontrollers. If you burn the security fuse of a device then nobody can read the contents of the device. If a device is programmed with an encryption code it can not be read unless the person operating the programmer enters the proper encryption code. Without this code only garbage data will be read.

Protection Fuses: Some FLASH based memory devices have protection fuses. A Protection fuse is used to protect a sector of memory in the FLASH device from being accidentally programmed or modified by the target hardware where the device will be installed. This fuse will need to be reset by a specific series of events before the data in the particular sector can be programmed or changed to new values. Protection fuses are only programmed using the separate "Protection" function or the "Auto" function. Please see the function descriptions section of the manual for more details.

3.1.2 Device Symbols for PLD, PAL, GAL, PEEL, CPLD, EPLD and FPGA

Programmable Logic Device (PLD): Generally speaking, a device that can be programmed to perform many different logic operations is a PLD. PLDs are generally grouped into following five categories:

PLD: A one time programmable logic device such as a PAL device.

EPLD: A UV erasable PLD such as an EPLD, CPLD, or FPGA device. These devices contain a transparent window on top for UV light exposure.

EEPLD: An electrically erasable PLD such as a GAL, PEEL, CPLD, or FPGA device.

CPLD: A more complex PLD device.

FPGA: Field programmable gate array.

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

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

Fuse Blown and Intact: A new programmable device generally comes with all fuses in an intact state (i.e. logic 1 or blank state). The designer can only blow an Intact fuse into a Blown state i.e. logic 0 state. The default state of a new device may be opposite to this depending upon the technology used to manufacture the device. For one time programmable (OTP) devices you can not change the state of a fuse back to the unprogrammed (default) state once it is blown (programmed). UV erasable devices have a window on them. When this window is exposed to UV light for some time the fuses will be erased to the default (or unprogrammed) blank state. Other devices are electrically erasable and can be erased by using an ERASE function on the Programmer.

Array Fuses & Configuration Fuses: Array fuses are the main logic fuses in a PLD. Different arrangements (Blown/Intact) will have different logic functions. Configuration fuses indicate the I/O architecture of the PLD, such as Combinatorial/Registered, Output Feedback/Output Enable. Generally the end user does not have to know the function of these fuses because the logic compiler used to translate your logic statements and equations into a JEDEC fuse map or other format for programming into a device will specify the value of these bits.

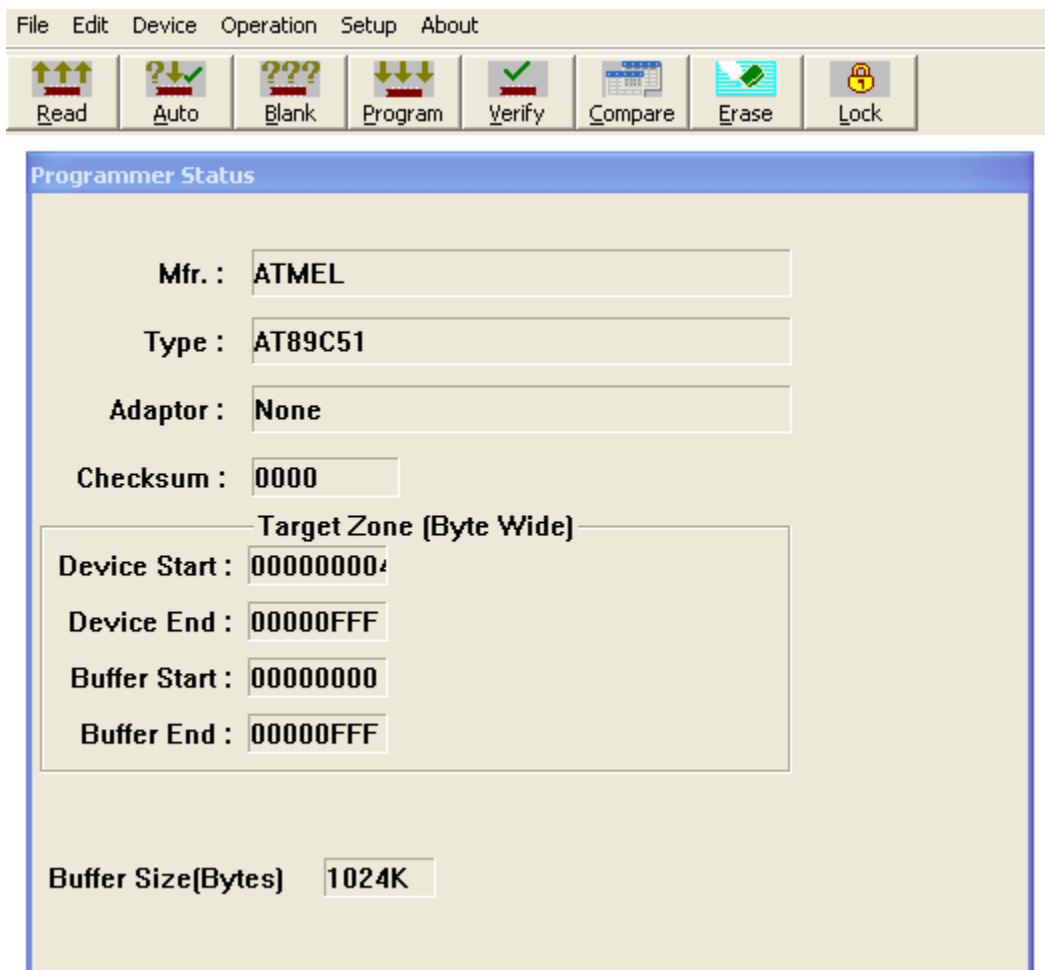
Security Fuses: Security fuses are present in many PLD/CPLD devices. When a security fuse is programmed the data stored inside the PLD can not be read by a device programmer for reverse

engineering purposes. Generally a device will read as blank if the security fuse is blown. The device will operate functionally equivalent with the security fuse intact or blown.

3.2 Viewing the Main Menu of a Device Driver file:

Before we begin operation please take a moment to familiarize yourself with the main functions found in the device driver files. Device driver files contain all of the device specific functions. Functions that are not available for certain devices will be faded out and can not be selected.

3.2.1 Main Function Menu All main programmer functions are listed on the left side of the window. Status information is displayed on the right side of window.



3.2.2 Status Field

In the upper right side of the window is status information about the target device being used including MFR, TYPE, Adapter, checksum. This is called the Device Status Field. Below the Device Status Field there is information about the target device being used, such as device start address and device end address, buffer start address, buffer end address, buffer size. This is called Buffer Status Field.

Whenever the programmer is being used, please make sure that data in the status field matches your requirements. Otherwise the target DEVICE may be damaged or programmed to an unknown state. Many devices have ID codes that the programmer will read to make sure the software algorithm called is the correct algorithm for the target device inserted in the socket. However, not all devices have an ID code so it is best not to rely on this feature.

By now you should be familiar with the main menu screen. A brief overview of most of the available functions is contained in the rest of this section. All relevant functions will be discussed in detail in later chapters.

3.3 Getting Started

The following example will demonstrate how to start and exit the device driver main menu through “UPMain” and use functions such as DEVICE, LOAD, BLANK CHECK and PROGRAM. If you do not get the desired results please check that your installation is correct.

In the example, we are going to use an AMD AmPAL16L8A PLD device. The procedure for working with other types of devices is similar.

3.3.1 Starting from the “UPMain” file

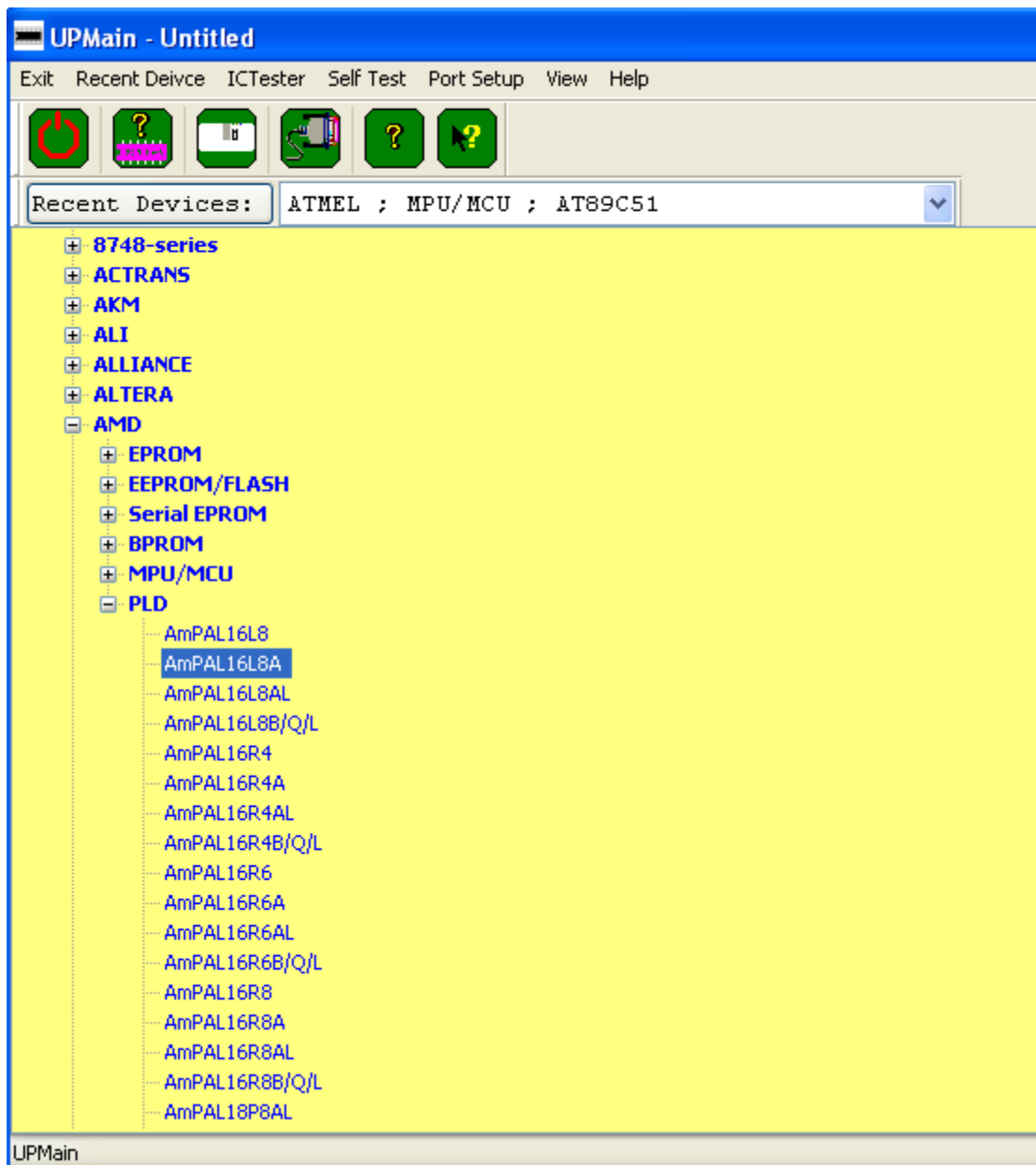
3.3.1.1 Execute the main program

Double clicking the UNIVERSAL PROGRAMMER & IC TESTER this will open the following window.

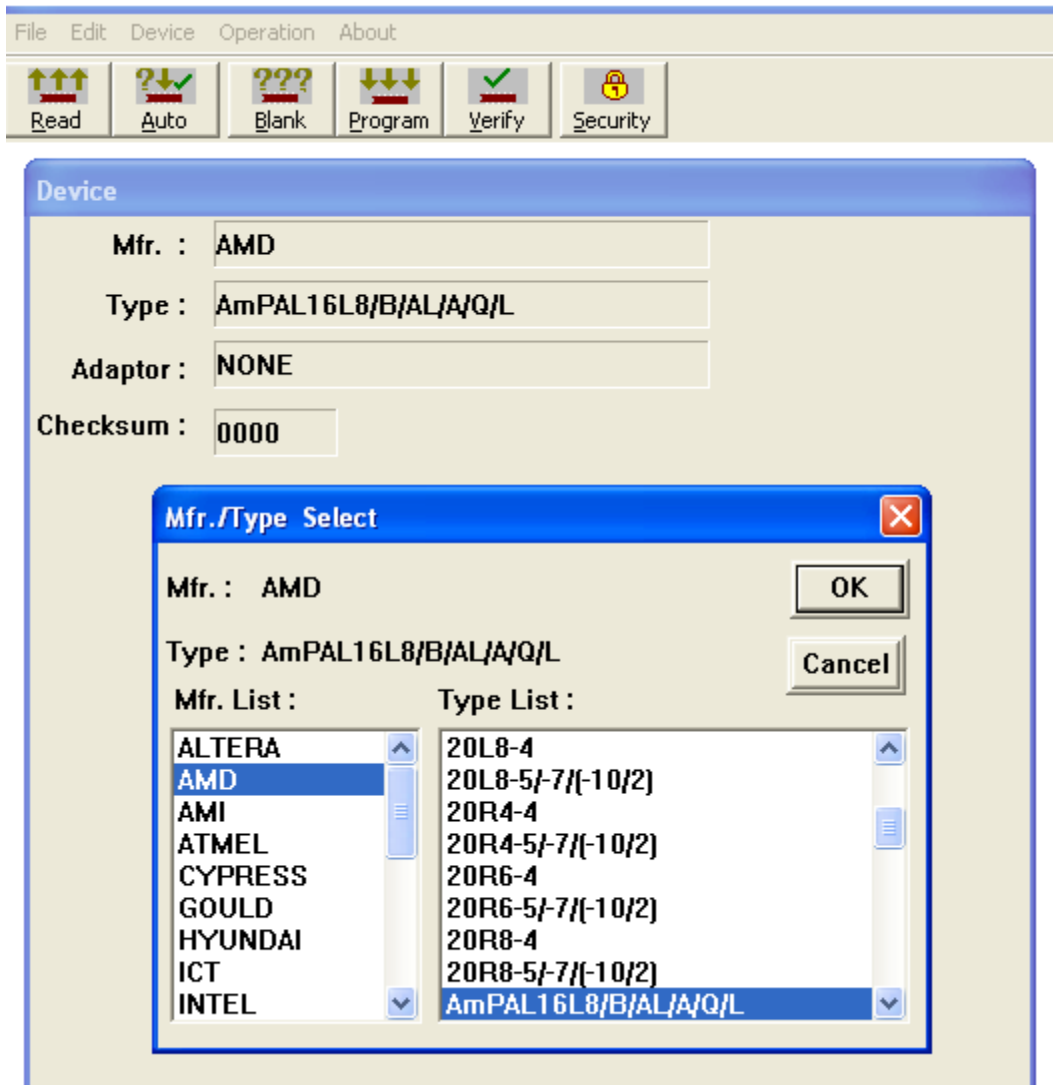


3.3.1.2 Select manufacture Device Group and Type

Select the desired manufacturer, in this case AMD, expand the tree to select Type (PLD) then expand the tree to select Device (AmPAL16L8A). After selecting the correct device, double click the selected device. UPMain program will automatically search for and execute the relevant device driver file for the manufacturer and type selected. In this case the WPLD1.EXE driver file will be executed and the status field of the driver file will indicate the device manufacturer and type that was just selected and the driver file main menu will be displayed. Please note the UPMain program will record each IC type you select so the next time you can use the Recent Devices button to select the desired device quickly.



Software Note: All driver files can be executed individually. You can also select PLD MANUFACTURER and TYPE directly within individual driver files. Select the "Device" function to get a list of device manufacturers and types supported by the current device driver file. Make a selection by moving the mouse to the desired manufacturer and clicking the left mouse button or by using the < Up > and < Down > arrow keys followed by pressing the < Enter > key when the desired device is highlighted. The WPLD1.EXE manufacturer and type selection window is shown below.

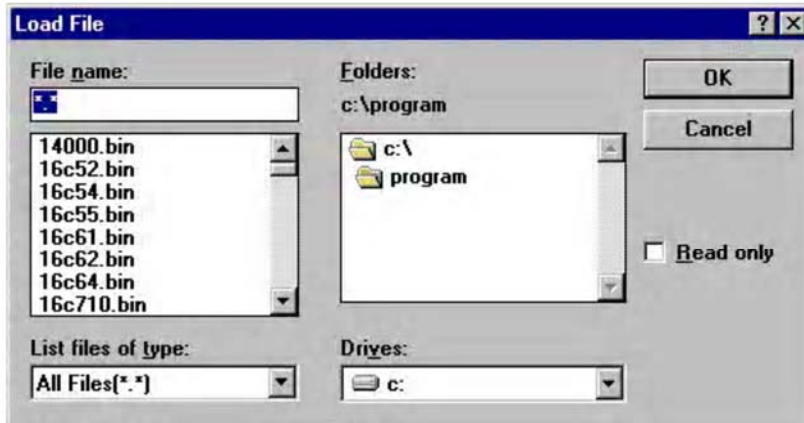


Troubleshooting:

When the manufacturer and type are entered, the data will be updated in the status fields and the relevant driver file will be loaded to the programmer docking station. If you select a device that requires an adapter to be programmed you may also get this message.

3.3.2 Load Disk File into Buffer

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



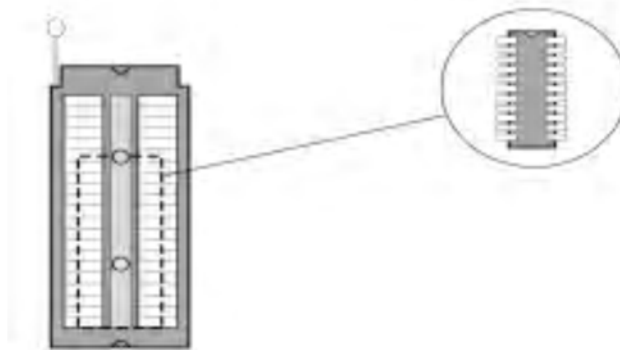
Submenu of file loading

The file loading window is similar to file loading/opening windows in most other windows programs. The desired filename may be typed in the "File: " input box and the indicated file will be transferred to the programmer's working buffer. The drive letter and any path names must be correct to get the desired results. An alternative to the direct typing method is to select a file from the files displayed in the list box directly under the "File Name:" heading. You can mask which files are displayed by clicking a file type mask in the "List Files of Type: " control box. To list files found in a different directory click on directory folders in the Directories list box shown on the right side of the window. To list files found on a different drive click the down arrow next to the "Drives" control box and select a new drive letter (i.e. a:, b:, e:, etc.) If you do not like to use the mouse you can use the <TAB> key to switch between windows and the <up> and <down> arrow keys followed by <enter> to make selections.

3.3.3 Read Contents from Master PLD

If the PLD data is in a MASTER PLD instead of a disk file, you can transfer the device contents to the working buffer by typing "R" or clicking on the "READ" button.

Insert the MASTER PLD into the socket while the lever on the 40-pin DIP socket is in the up position. Pin 1 should be located on the lever side of the DIP socket. If the device has less pins than the DIP socket then the device should be bottom justified in the socket as shown in the following diagram:



Close the lever. Then click the "OK" button or press 'Y' on the keyboard. The data of the MASTER PLD will be transferred to the programmer buffer and the window will display:

Reading now...

OK!

Click the "Cancel" button or press < Esc > to return to the main menu.

3.3.4 Insert a Blank PLD into the Socket

After transferring the data from the disk file or the MASTER PLD to the working buffer you are ready to program a new device. Insert a blank PLD into the DIP socket on the programmer while the pull lever is up. Again, Pin 1 should be located on the lever side of the DIP socket. If the device has less pins than the DIP socket then the device should be bottom justified in the socket. Close the lever. (Please see the previous diagram.) If you are not sure if the device in the socket is BLANK please use the "B" function to test if the device is really blank before programming.

CAUTION The notch-end of the PLD must face the lever on the programmer DIP socket. If the device has less than 40 pins then the device should be bottom justified. If the IC is inserted incorrectly the PLD may be damaged or programmed to an unknown state. As mentioned earlier, the programmer will read an ID code in the device to make sure it is talking to the correct device and the device is in the correct position but not all of devices have an ID code so it is best not to rely on this feature.

3.3.5 Program Buffer Contents to PLD

After loading the disk file or reading the MASTER PLD data into the working buffer and inserting a blank PLD, the program function is used to copy the contents of the working buffer to a device. Invoke the programming function by clicking the "Program" button or typing "P", the PROGRAM function will open the window:

Move the mouse pointer to click on "OK" or press "Y" on the PC keyboard to start programming the buffer contents into the blank PLD. After programming the device, new contents will be compared against the contents stored in the programmer's working buffer. This is called a VERIFY operation. If the contents match then the DONE LED will light. For programming other PLD's, wait until the BUSY LED turns off, then replace the PLD in the socket with a new blank device and click "OK" again or press 'Y' on the keyboard.

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

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

3.4 Expansion Adapter and Converters

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

3.4.1 Adapter and Converter Installation

Adapter – Each adapter has a 40-pin DIP footprint and can be plugged directly into the 40-pin DIP socket on the programmer. Adapters support devices that have 44-pin or more or devices that require some special programming requirements.

Converter- Each converter translates from a DIP footprint of a device to a surface mount footprint (PLCC, SOP, TSOP, etc....). Converters support devices with 8-pin to 44-pin

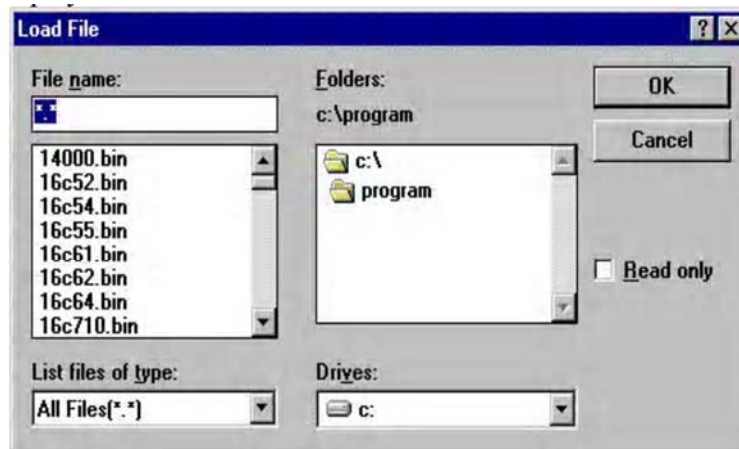
H/W Installation – Insert the adapter or converter into the DIP socket of the programmer just like you would install a 40-pin DIP IC into the socket as shown in the following figure. Some converters have a DIP footprint smaller than 40-pin and should be bottom justified just like an IC with fewer than 40 pins. Converters generally have a notched corner on the bottom PCB that indicates the position of the DIP footprint Pin 1. Adapters have a Pin 1 marked on the top PCB or a silk screen drawing of DIP IC with a notch in the drawing to indicate Pin 1 of the DIP footprint. The silk screen is found on the bottom PCB facing upwards.

4. FUNCTION REFERENCE GUIDE

The following reference guide describes each of the functions found in the main menu of the Device Driver files. The functions will be described in the order given in the menus. To enter into the desired function, type the first letter of the function name or click the left mouse button when the desired function is pointed to with the mouse pointer. Software Note: All software drivers basically contain 1 of 2 screen layouts. Memory devices and microcontrollers use one layout and PLD/PAL/GAL/CPLD and other programmable logic devices use another screen layout. Each layout contains all of the functions relating to those particular family of devices. If a device does not support a particular function then that function will be displayed as faded and can not be selected.

4.1 Load BIN or HEX File to Buffer

Select "Load file to programmer buffer" from the "File" menu. A dialog window will be displayed on the screen as follows:



<Load file>

The file loading window is similar to file loading/opening windows in most other windows programs. The desired filename may be directly entered in the File: input box and the indicated file will be transferred to the programmer's buffer. The drive letter and any path names must be correct to get the desired results.

An alternative to the direct typing method is to select a file from the files displayed in the list box directly under the "File Name:" heading. You can mask which files are displayed by clicking a file type mask in the "List Files of Type:" control box. To list files found in a different directory click on directory folders in the Directories list box shown on the right side of the window. To list files found on a different drive click the down arrow next to the "Drives" control box and select a new drive letter (i.e. a:, b:, e:, etc.) If you do not like to use the mouse you can use the <TAB> key to switch between windows and the <up> and <down> arrow keys followed by <enter> to make selections.

File Format Window

Once the desired input file is entered or selected move the mouse to the "OK" button and click the left mouse button. Once the file is entered, the window will automatically switch to the "file Format" window as shown in the next diagram. The following file format choices are available for most devices:

1. BIN file
2. INTEL HEX (for INTEL Hex file)
3. MOTOROLA S HEX (for Motorola S record formats)
4. JEDEC (for PLD only)

You should select 'Binary' if the file you are loading is a binary file, 'Intel HEX' if the file you are loading is an Intel HEX file, or 'Motorola S Record ' if the file is in one of the Motorola S Record formats, and 'JEDEC' if the file is in standard JEDEC fuse map format. If your assembler or compiler has the option to output binary files it is the best to use that option when loading files for memory devices or microcontrollers. Files already in binary format will be loaded faster. All HEX files are actually converted to Binary by the driver file before being loaded to the buffer for programming.

File Status Settings:

For BIN or Hex formats you can key in the file start, file end address, and buffer start address to specify where to load the file. The default for most device types is to load your entire file to the buffer and to locate the data at buffer address 0000h. To do this, do not change any values in the file status area. The following is a description of other possible File Status settings:

File Start: To load a file starting at the first data location in the file, leave this value at 0000h. If you only want to load data beginning at a certain address, 2000h for example, you would enter 2000h for this value.

Troubleshooting: If you leave this value at 0000h and load a HEX file you may get the message “ Some HEX codes are out of range ” and you may get a checksum of 0000h or F800. This means that none or only a portion of your data was loaded to the buffer. HEX files contain address information and may contain an offset. For example, if you load a HEX file with an offset of 1000h to the buffer while leaving the File Start value as 0000h and the Buff Start value as 0000 then buffer locations 0 -> FFFh will be filled with your Unused Bytes value and your file data will appear starting at location 1000h. Please consult your assembler or C compiler linker manual for details on output file formats.

File End:

This is the last address of the file being loaded that will be placed into the programmer's buffer. If you want to load an entire file then leave this value at its default value (don't change this value in this case). If you only want to load file data from the file start address up to an address before the end of the file, 1000h for example, you would enter 1000h for this value.

Buffer Start:

This is the address in the programmer's buffer where the file data located at the file start address mentioned above will be placed. The default value for this option is 0000h. If you want the file data to be placed in the buffer starting at a different location then enter that address here.

Unused Bytes Settings:

If “ Intel HEX ” or “ Motorola S Record ” formats are loaded, you can choose to fill 'Unused bytes' with '00' or 'FF' or '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 data. The rest of the unused portions of the EPROM may be filled with all 00's or all FF's depending on your preference. If you would like to load one or more additional HEX files to one DEVICE then you should choose 'Don't care' for the Unused Bytes option. This means that the portions of the buffer not used by the HEX file being loaded to the buffer will not be changed in any way. For example, if you load one HEX file and then load an additional HEX file to the same buffer and select ' 00 ' or ' FF ' for the unused bytes option

then all portions of the buffer that are not filled by the new file will be filled with ' 00 ' or ' FF ', including the areas where the first file you loaded is occupying. To get the correct results choose the " Don ' t Care " option for the Unused Bytes selection when loading additional HEX files to the buffer.

4.2 Save Programmer Buffer to Disk

Click on "Save File from Programmer buffer" from the "File" menu. A dialog window will appear.

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

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

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

4.3 Edit Buffer

Click "Edit" and select "Edit buffer" and the following window will be opened:

Editor																
Fill	Jump	Move	Swap	CalcSum	Search	Print	Exit									
ADDR	0	1	2	3	4	5	6	7-8	9	A	B	C	D	E	F	0123456789ABCDEF
000000	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
000010	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
000020	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
000030	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
000040	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
000050	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
000060	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
000070	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
000080	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
000090	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
0000A0	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
0000B0	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
0000C0	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
0000D0	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00

The editor is used to view or modify data contained in the programmer buffer. The following functions are available in the editor:

- 1. Fill:** To fill a block of addresses with the same value.
- 2. Move:** Copy data from one block of buffer memory to another block of memory.
- 3. Swap:** Exchange the high and low nibbles of each byte in the buffer.
- 4. CalcSum:** Calculate the check sum for a block of memory.
- 5. Search:** Search the programming buffer for specific ASCII or binary data.
- 6. Exit:** Return to the device driver file menu that called the editor.

All of these functions bring up small parameter windows where appropriate. Parameters include start and end addresses to perform functions on and data values to fill the buffer with and values to search for.

4.4 Modify Programmer Buffer Mapping

The modify programmer buffer function is used to select what portion of the buffer is programmed to a target device and what portion of the buffer is included in the checksum calculation. Usually this function is used to limit the size of the buffer if a file is small compared to the EPROM size of the target device you will program. Click on “ Setup ” in any driver file and then click on “ Modify Programmer Buffer Mapping ” and the following window will appear:

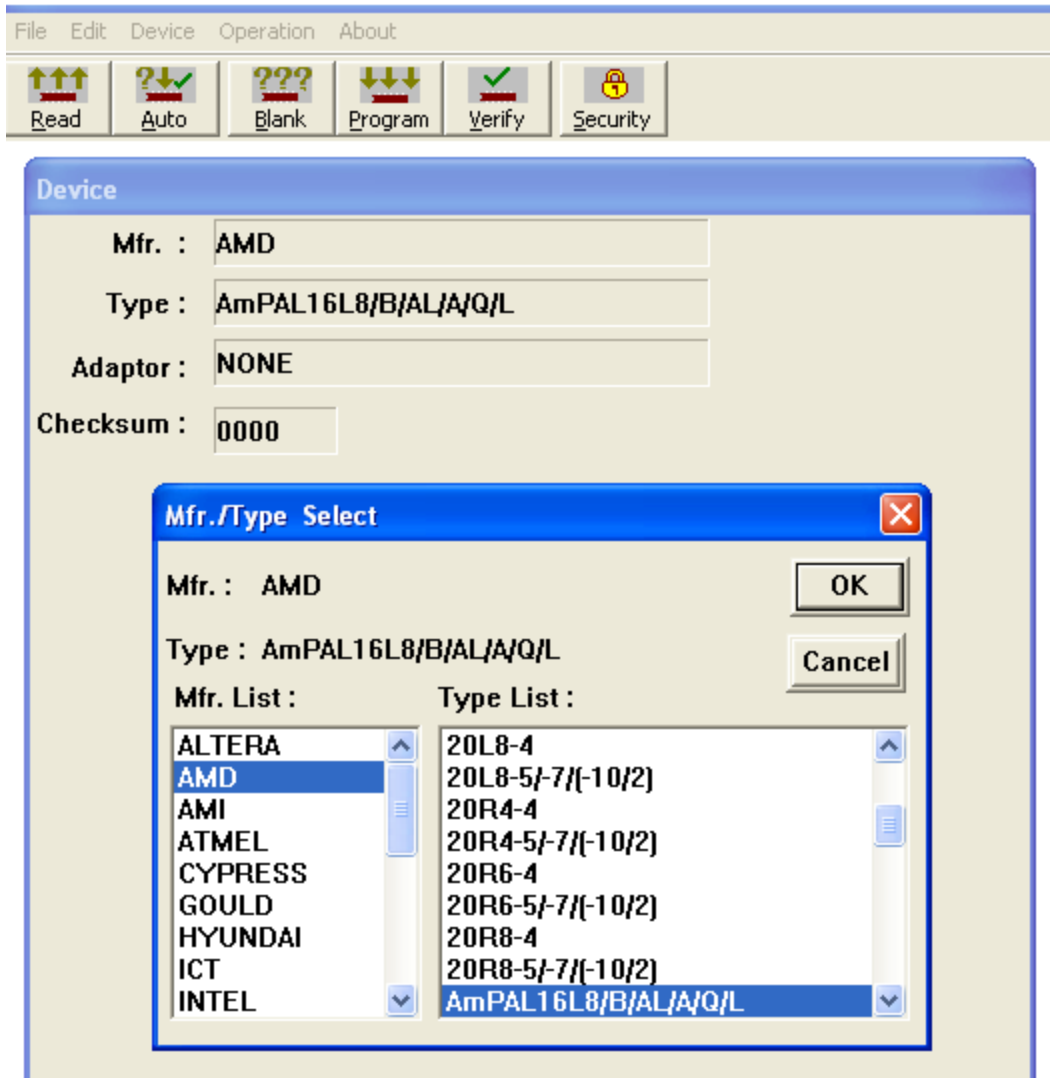
To modify the buffer mapping type new values in for the device start and end addresses and for the buffer start and end addresses. Addresses should be entered as HEX values. The “ Device: ” start address is the address in the target device where the information starting at the “ Buffer: ” start address will be programmed to in the target device. The values entered in HEX code will be updated in the STATUS FIELD for reference.

Note: When the type or manufacturer (Mfr.) selection is changed the buffer mapping will default to the normal mapping specified by the device selected. The normal address range is equal to the entire address range of the target device. This is not supported by all driver software to ensure correct data programming.

Click "Cancel" to return to the main menu.

4.5 Device Select

Click on "Device" and the following screen will appear:



A list of manufacturers is displayed on the left side of the screen under the “ Mfr. List: ” heading and a list of device types for each manufacturer that is selected is displayed on the right side of the screen under the “ Type List ” heading. Once a desired IC manufacturer and device type are selected click the “ OK ” button and the STATUS FIELD will be updated with the new manufacturer and type selection.

Click < Cancel > to return to the main menu.

Troubleshooting: When the manufacturer and type are entered, the data will be updated in the status fields and the relevant driver file will be loaded to the programmer docking station. If the message "BIN not found" appears that means that the software for the selected manufacturer and device type is not installed on your PC. Please check your installation diskettes for the requested file or contact your local distributor. If you select a device that requires an adapter to be programmed you may also get this message.

4.6 Exit

Click on Exit is to leave the current function and return to windows. Before exiting the software will save parameters including manufacturer and Type.

4.7 Blank Check

Click "Blank check", and the Blank check window will be displayed.

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

4.8 Read

Click "Read" and the Read window will be displayed:

Click the "OK" button or press " Y " on the keyboard to read data from a device in the socket to the programmer buffer. Data will be read from the device according to the settings in the Programmer Buffer Mapping status window shown in the mapping STATUS FIELD. Click "Cancel" to return to the main menu. The Read Window will display the " Reading now... " message, and when the read is completed, the "OK!" message will be displayed and the DONELED under the socket will light.

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

4.9 Verify

After clicking on "Verify", the verify window will be displayed:

Click the "OK" button or press " Y " on the keyboard to verify that the target device contents match the contents stored in the programmer buffer at the address range shown in the STATUS FIELD, or click "Cancel" to return to the main menu. The Verify Window will display the "Verifying now" message. When the verify function has been completed, the " OK! " message will be displayed on the screen and the DONE LED below the socket will come on.

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

4.10 Program

Click "Program". The program window will be displayed.

Click the "OK" button or press " Y " on the keyboard to begin transferring the data in the working buffer to the target device, or click "Cancel" to return to the main menu. The " Programming now " message will appear on the screen and the counter in the Status Field will increment until the function is completed. Then, a " Program OK! " message will be displayed on the screen and the DONE LED will come on if the operation was completed successfully.

4.11 Auto

Click "Auto". The Auto window will be displayed:

Click the "OK" button or press " Y " on the keyboard to start the Auto function, or click "Cancel" to return to the main menu. There are many choices to be enabled or disabled in the auto function. The Auto function window displays all available functions for a given device manufacturer and type. Next to each function is a box that can be enabled (checked) or disabled (left blank) by using the mouse. Boxes filled in with a solid gray color can not be selected. This means that the particular device that is selected does not support the displayed function.

4.12 Security, Lock Bits & Encryption Code

Some microcontrollers have special facilities such as Lock Bits, Security Bits, or Encryption Codes to prevent unauthorized copying of proprietary code. The user can easily operate these functions by selecting " L ", " S ", or " E " under the main menu, or use the " Auto " function to program the Lock Bits, Security Bits, or Encryption Code. Refer to each individual microcontroller ' s manual for the definition of these special options as their definition may change slightly from manufacturer to manufacturer and even from type to type from the same manufacturer.

4.13 LEDs

The three LEDs on the programmer module are labeled ON, BUSY, and DONE. The ON LED will light up after turning on power switch. The BUSY LED will light up during device programming, reading, and verifying. The DONE LED will light up when the result of an operation is positive.

About PLD (PAL, GAL, PEEL, & EPLD)

The functions used for PLD programming are almost the same as those used to program memory devices and microcontrollers. The following section describes the additional functions.

4.14 Additional or Alternate Functions for PLD Programming

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

Save JEDEC File: Under PLD Device Driver files, data is saved in JEDEC file format instead of binary format. Files are saved in other formats where are appropriate.

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

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

Security Fuse Blow: Click "Security" to blow the security fuse of a PLD device. This is the final step in PLD programming. Blowing the security fuses can prevent any further access to the PLD either for "modification" or for "reading". Blowing the fuses prevents anyone from making unauthorized copies of your PLD.

NOTE: After the security fuse is blown the device data can not be read or verified.

4.15 Save Project:

This feature allows you to save all settings selected in the software driver, including:

1. Mfr. & Type selections
2. Load File setting
3. Target Zone settings including Device Start, Device End, Buffer Start, & Buffer End
4. Other programming settings
5. Auto function settings

After all settings have been entered and the Auto Function section completed, select the Save Project under the “ File ” dropdown menu and type the filename you wish to save it under.

4.16 Open project

After saving the configuration information settings using the Save Programmer Configuration command, you may then use the Load Programmer Configuration to load the desired configuration into the software driver. You select the Load Programmer Configuration under the “ File ” dropdown menu and then choose the filename of the configuration data you wish to use. The software will automatically load the programming data information (note: not all software contains Target Zone data). The “ Auto ” function will also reinstall all the settings from the previously programmed device. After the Open project window has been completed, users only need to select the "Auto" function to begin programming.

4.17. SERIAL NUMBER FUNCTION (SERIALIZATION)

1. Press S/N Setup button in the “ AUTO ” window to setup the Serial Number function

2. Serial Number On/Off:

This function can be selected to enable or disable the Serial Number function (The Serial Number function is only available in the “ AUTO ” mode.)

3. Length:

This function sets the length of the serial number. It can be up to 8 bytes (16 digits) long.

4. Display Format:

This function selects the display format for the serial number, either HEX (Hexadecimal) or BCD (Binary Coded Decimal).

5. S/N Start Address:

This function is used to set the starting write location on the ROM.

6. Start Serial No.:

This function is used to set the starting value of the serial number.

7. Direction:

This function is used to specify the sorting order of the serial numbers either from low to high (Low Byte) or high to low (High Byte).

8. After programming the data, the serial number will automatically increase itself.

9. Not all the programmer software can use the Serial Number function (ex: PLD software can not support this function.)

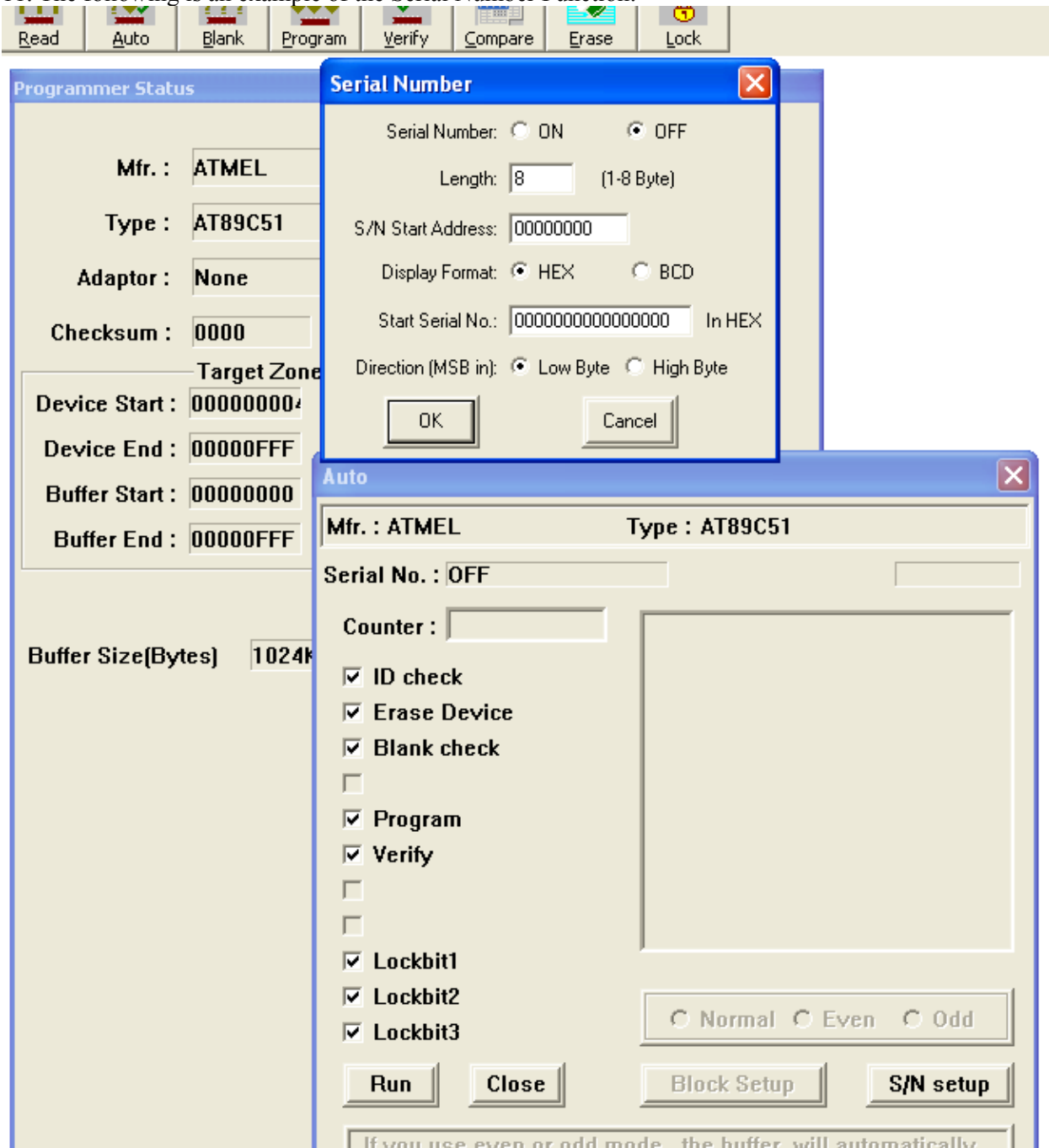
10. The serial number setting for Microchip PIC series devices is different from other devices. Its high byte serial number location 's data will be fixed (12 Bits device is 08H. 14 Bits device is 37H, 16 Bits device is B6H). For example, if the user selected serial number is “ 1234ABCD ” then data will be programmed from the beginning of the serial number start address and its data will be formed as follows:

Programmed data to PIC16C54 (12 Bits) is 0812H 0834H 08ABH 08CDH

Programmed data to PIC16C62 (14 Bits) is 3712H 3734H 37ABH 37CDH

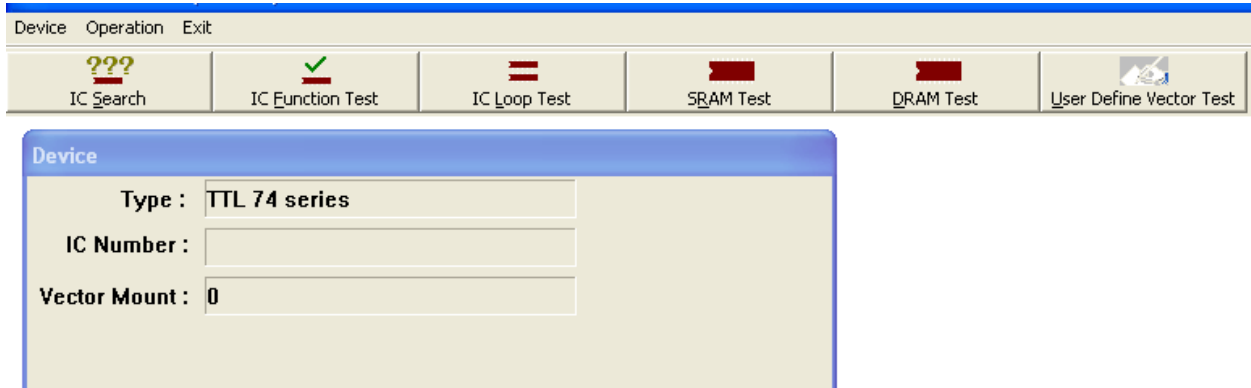
Programmed data to PIC17C42 (16 Bits) is B612H B634H B6ABH B6CDH

11. The following is an example of the Serial Number Function:



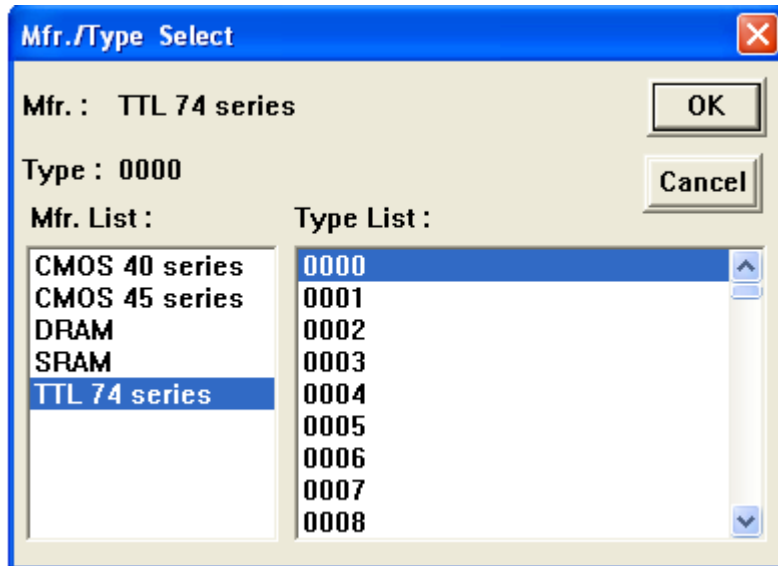
5. IC Tester:

This chapter describes IC Testing procedure and also the different functions available to the TESTER menu .



5.1 Device

Click on "Device" and the following screen will appear:



The required IC Type and number can be selected and click <OK>

5.2 Operation

- IC Search
- IC Function Test
- IC Loop Test
- SRAM Test
- DRAM Test
- User Define Vector test

5.2.1 IC Search

The unknown IC will be compared with the currently selected library, if it is not found a message to that effect will be displayed. As the parameters of an unknown chip may be contained in any other libraries, it may be necessary to search all others in turn. If a library IC or ICs meeting the unknown IC's logic is found, then the IC number or numbers will be displayed.

5.2.2 IC Function Test

This is a single loop function test that is the possible logic combinations are carried out once only

5.2.3 IC Loop Test

This option will repeatedly test the chip logic combinations in an endless loop. In the event of an error during any of the tests, testing will stop and an error message will be displayed. The test may be terminated by pressing <Esc> button.

5.2.4 SRAM Test

This option is used for Static RAM test.

5.2.5 DRAM Test

This option is used for Dynamic RAM test.

5.2.6 User Define Vector test

This function allows any logic input, expected output, VCC and GND to be defined on the 24 pins of the test socket. You may write your own vectors to test any of sequences of logic steps for new ICs including PAL, CPLD, and PROM etc. The test vector syntax is described in detail below.

Test vector Syntax

A test vector is the definition of input and output logic states applicable to the various Pin of an IC. Up to 500 vectors may be specified for a single IC thus allowing a sequence of logic events to be tested.

A: Active vector

D: Disable vector

0: Apply Low to IC

1: Apply High to IC

L: Expected output result from IC is Low.

H: Expected output result from IC is High.

N: Same symbol as last vector.

X: Same symbol as last vector, but don't care output result from IC.

E: Apply 5V to IC, Only Pin 9, 19, 20, 22, 24 can be specified.

G: Apply GND to IC, only Pin 12, 16 can be specified

<Space>: Apposite status.

Note: If for example a 24 pin IC is to be tested, then ZIF socket pin 9 is electrically equivalent to pin 1 of the IC and so on.

For example 7404 IC can be tested with the following

Pin No. 111111111122222

A/D 1 2 3 4 5 6 7 8 9012345678901234

A V001 XXXXX0H0H0HGH0H0H0EXXXXX

A V002 XXXXX1L1L1LGL1L1L1EXXXXX

A vector can be executed by selecting the function test or loop test.

The test sequence is as follows.

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

APPENDIX

TROUBLESHOOTING

For problems with individual functions please refer to the specific function description under the Function Reference Guide.

PROBLEM 1

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

RESOLUTION

1-1 You may not have the serial port set correctly for your programmer. Double check the port Setup assignment.

1-2 There may be a bad connection between the RS232 port and the programmer or you may have a faulty cable. Double check the cable connection.

1-3 You may have the baud rate set too high for your PC's serial port hardware.

PROBLEM 2

You have two or more programming failures in a row.

RESOLUTION

2-1 Check with your distributor to make sure you have the latest driver file versions.

2-2 Check if the device is inserted in the ZIF socket correctly

Note:

When you install the software in WINDOWS NT4.0 or Above, you must be logged on as an Administrator first.

Things to do before calling your supplier

1. Reboot the computer and try again.
2. Repeat all the steps, following the instructions in this manual.
3. Close all other programs running under windows.
4. See if your problem is listed in the Trouble-Shooting section.
5. Try it on another system.
6. Compare system requirements with your configuration.
7. Ask your in-house specialist (every office has one).
8. Consult the person who installed the product if available.

General Trouble shooting checklist

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

1. Is the serial port interface card fully seated in its slot?
2. Are all cable connections securely fastened?
3. Is the setting on the RS-232C port correct?