LV40 *Portable*

Manual for PC and Stand-alone Operation

Contents

Introduction

Congratulations on choosing the ICE Technology LV40 *Portable* Universal device programmer. We are sure that you will find the programmer and PC software very easy to use and full of features to support all of your device programming requirements, whether you are using it for development, field service, or low-medium scale production. With your new programmer you will be able to

- ✓ Read data from programmed devices
- ✓ Load data in from disk
- ✓ Edit data
- ✓ Programme a device
- ✓ Verify the contents of a device

There are many other features of your programmer supported via the easy to use menu driven PC software as well as in Stand-Alone mode.

About this manual

This manual provides the instructions for using the LV40 Portable programmer, both in PC-based and Stand-Alone modes. The PC software is identical to that of other ICE Technology single socket programmers, and indeed can operate from the same software. In Stand-Alone mode, the LCD display on the programmer mimics the menu facilities of the PC software, which adds to its ease of use. The manual is divided into two main sections, the first describing the PC software, the latter describing the Stand-Alone use.

The manual is laid out so that is possible to programme a device quickly by following one of the *QuickStart* tutorials. A reference section follows which gives instructions in greater detail.
It is recommended that you read through the relevant sections of this manual before using the programmer, and in case of any difficulties encountered during use.

Conventions

Throughout this manual, certain conventions will be followed in typefaces in order to make instructions clear.

e.g. In the line:

Type `EPROM <ENTER>`

`EPROM:` this font is used throughout the manual to indicate something that should be typed into the computer. If this is encased in[ ], then it is the option to be selected from the programmer display:

`[EPROM] <ENTER>`

`<ENTER>:` Any items encased in <> indicate the name of a key to press, i.e. in this case the `ENTER,` or ↵ key.

Menu selections are indicated by words separated by forward slashes, e.g. `FILE/LOAD` indicates select the menu option `FILE,` followed by the sub-menu option `LOAD.`

Messages displayed by the programmer software are indicated in italics, for example:

DEVICE PROGRAMMED AND VERIFIED

Contents of the package

When you open the package you will find :

- LV40 Portable universal device programmer
- A floppy disk containing the software for the programmer.
- User manual
- A power supply (UK, Euro, USA or Japanese)
- A D25 male - male parallel cable
- Free software may be included in the package at the discretion of ICE Technology. These packages may assist you in product development. Please see the READ.ME files on the individual disk(s) for information on how to use these packages and whom to contact for further information.

The programmer can be operated from batteries. These are not included as standard. Use 8 alkaline AA cells (standard or rechargeable).

Low Voltage Features

The LV40 supports Low-Voltage devices, including 1.8V and 3.3V parts, as well as standard 5V devices. It is capable of programming and verifying devices down to 1.8 Volts. When using low voltage devices, the programming algorithm in the system will automatically be set to the correct voltages as specified by the manufacturer's programming algorithm.

Operating from Batteries

The LV40 can be operated from mains or batteries. To operate from batteries use 8 alkaline AA type cells, either standard or rechargeable. Batteries can be recharged in system, with a utility provided within the firmware to activate the recharging. The battery cover is held in place with screws.

WARNING : USE ONLY THE SCREWS PROVIDED TO FASTEN THE BATTERY COVER, & DO NOT OVER TIGHTEN THE SCREWS.

NOTE: Certain Bipolar devices require high currents in very short bursts and therefore some batteries may not always be able to handle these parts.

Device Position

Figure (i) below illustrates the orientation of chips in the programmer. The chip is located at the bottom of the ZIF socket with the bottom pins next to the ZIF socket handle. There are just a few exceptions to this rule, which will be advised by the programming software.

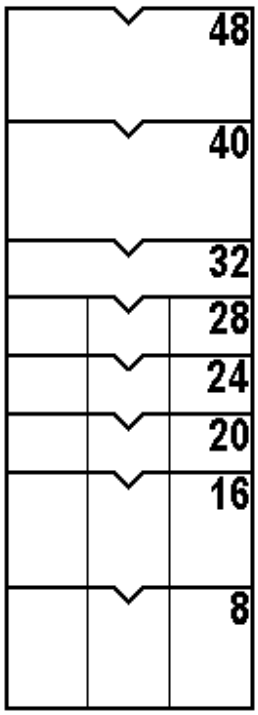

figure (i)– Positioning the device to be programmed in the programmer ZIF socket

If you are using a socket adapter, please see the instructions below:

Using non-DIL devices and parts with over 40-pins

Devices in other packaging than dual-in-line, and devices over 40-pins require socket adapters. A range of adapters is available from ICE Technology for most common parts. If you are unsure as to which adapter to use, please contact ICE Technology or your distributor for advice. A full list of current adapters is available from the ICE Technology Home Page on the Internet. See Appendix D for details.

Adapters from other sources that are simple pin conversions can usually be used successfully. However, for devices over 40-pins, pin mappings are not usually standard, and it is therefore necessary to use an adapter specifically designed for the programmer in use. Where devices require non-standard adapters to be supported, these are available from ICE Technology.

All adapters produced by ICE Technology are standard across the whole programmer range.

Adapter usage

Insert the adapter into the ZIF socket of the programmer with the bottom end of the adapter in the bottom pins of the socket. The text on the adapter boards should be the right way up. Lock the ZIF socket handle in place. For PLCC adapters, the notched corner of the adapter socket will usually be in the top left hand corner. Devices are placed in the socket in the ‘live-bug’ position, i.e.. with legs downwards for all sockets 28-pins and above. 20-pin sockets are “dead-bug” loading. Push the device down to lock in place, then push down the socket sides to release.

See figure (ii) below.

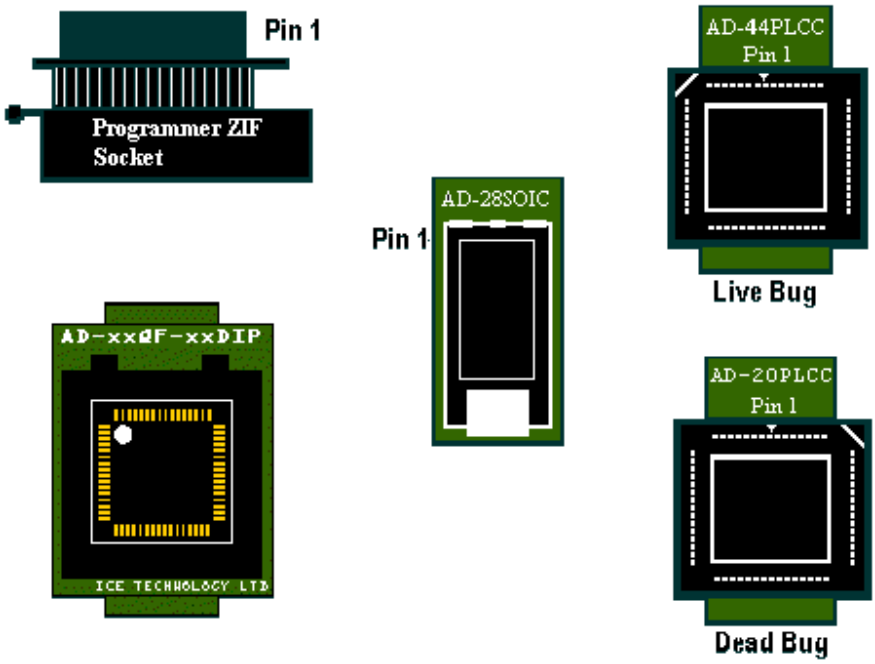

figure (ii) - Adapter Usage

SECTION I

Using the
LV40 Portable
Programmer on a PC

1. Installing the Software

The LV40 is shipped ready for use. Internal memory and firmware contain all the software libraries and algorithms to enable the full range of memory, programmable logic and microcontroller devices to be programmed without the need of a PC. See Section II for full details. However, we recommend that users become equally familiar with the PC software to fully appreciate the operation of the programmer. Software is provided on 3.5" floppy disk and also provides utilities to enable the programmer firmware to be updated via either the parallel or serial port of any PC.

1.1 Latest Software

If you have purchased your programmer through one of ICE Technology's distributors, it is advisable to contact ICE Technology to ensure that you are running the latest version of software. New software versions are released on a regular basis, and should the distributor hold stock for even a relatively short period of time, the software may already have been superseded. Contact information is given in Appendix D. of this manual, including details of our download service and Internet Web site, where software is available 24-hours a day.

1.2 Making a backup

Before running the software it is advisable to make a back-up copy as soon as you receive the disks. As only the used sectors of the disk are formatted then `DISKCOPY` will not copy the disk correctly.

Use: `COPY A:*.* B:`

1.3 Running the Install Programme

An install programme is provided on the floppy disk. This software copies all of the necessary files onto your hard disk. We suggest that you use the following directory to hold the programmer software

```
C:\ICETECH\
```

Where C: is the name of your hard disk

So, if you are installing the software onto the C: drive, to run the installation programme place the floppy disk in the drive and log onto this drive by typing **A:** or **B:** as necessary. Now type:

```
INSTALL C:\ICETECH <CR>
```

After installation you should add this directory to your current PATH statement in your AUTOEXEC.BAT file. For example change

```
PATH=C:\DOS;C:\BAT;C:\UTILS
```

to

```
PATH=C:\DOS;C:\BAT;C:\UTILS;C:\ICETECH
```

Then reset the PC.

2. Connecting the Programmer

2.1 Computer Requirements

The programmer will operate with any IBM compatible PC with a standard IBM CENTRONICS interface (parallel printer port). This means that it is very easy to move it between PCs. Users who prefer to have an internal card or who do not wish to use their existing parallel port can fit an additional parallel card (these are available from ICE Technology). Alternatively, a manual switch box may be used.

The software is DOS based and compatible with MS-DOS v.3.0 or later. It works happily in a DOS-Prompt window under MS-Windows, v.3.1 or Windows 95. Full Windows software, including Windows NT support, is under development

2.2 Connection to the PC

STEP 1 - Connect one end of the parallel cable to the programmer. The other end of the cable should connect to the PC. The cable will only fit the centronics port. Tighten the holding screws thus ensuring that the cable will not cause programming problems.
STEP 2 - Connect the power supply provided to the programmer and plug into any mains socket. Alternatively, fit 8 alkaline AA batteries.

USE ONLY THE POWER SUPPLY ADAPTER PROVIDED.

Replacements and adapters for different countries (UK / European / US / Japan) can be supplied by ICE Technology or their distributors.
STEP 3 - Switch on the PC.
STEP 4 - Run the SELFTEST programme.

2.3 Running the SELFTEST Programme

The software contains a diagnostic programme, SELFTEST.COM, which can check that the programmer has been installed correctly, and test the functionality of the programmer hardware. Before using the programmer for the first time, whenever moving it to a new PC, or if any fault is suspected, run the **SELFTEST** programme. This has two purposes :

-It confirms which parallel port is being used by the programmer.
-It checks most of the programmer hardware.

WARNING :

ALWAYS ENSURE THAT THERE IS NO CHIP IN THE PROGRAMMER BEFORE RUNNING SELFTEST, AS LEAVING A DEVICE IN THE SOCKET COULD DAMAGE THE CHIP AND GIVE FALSE RESULTS.

To run the programme, connect the programmer to the PC and either mains or battery power, remove any chips and type

SELFTEST <CR>

at the DOS prompt.

The SELFTEST will report the programmer's Firmware version number and run a communications test. It will then run through a series of hardware checks and will report any faults found. The programme will report on which port the programmer is connected to.

If an error is reported it is possible to obtain a printout by using

```
SELFTEST >PRN
```

where PRN is the name of the resident printer device (i.e.. LPT1).

or `SELFTEST >ERROR.LST`

followed by `PRINT ERROR.LST`

Should the Selftest fail for any reason, please contact ICE Technology. See Appendix D for Contact information

3. QuickStart Tutorials

We have provided some Quickstart tutorials to help you start using the programmer as quickly as possible. If you have not used a programmer before, these may help you to begin. However, these should not be taken as a comprehensive guide to the programmer's capabilities. The following tutorials are provided:

TUTORIAL 3.1 Programming an EPROM or Microcontroller from a master device.

TUTORIAL 3.2 Programming an EPROM or Microcontroller from a file on disk.

TUTORIAL 3.3 Programming a PAL or GAL from a file on disk.

These tutorials are provided to get you using the programmer quickly for simple tasks. However, we recommend that you read the reference sections of the manual if you have any difficulty and before attempting any more complex operations.

TUTORIAL 3.1:

Programming an EPROM or Microcontroller from a master device.

1. Type EPROM <RETURN>
2. Select the part you are using as a master, using PART/TYPE. Select the device type you are wishing to use and press <ENTER>. You will then be given a list of manufacturers who support the selected device type. Select the appropriate manufacturer and press <ENTER>. You will then be prompted for the part name. Choose the correct part name from the menu you are given and press <ENTER> to accept. The details of the device you have chosen will now be displayed at the bottom of the screen.
3. Ensure that the buffer size is large enough to hold the data. You can change the buffer size using BUFFER/SIZE. See section 5.2.1 for more information.
4. Insert the master device in the programmer.
5. Select ACTION/READ to read the contents of the chip into the buffer. Press <F10> to accept the default addresses.
6. Take the Master out of the programmer.
7. Select the part you are copying to if it is different (see 2.)
8. Place the copy device in the programmer.
9. Select ACTION/AUTOPROGRAMME. Press <F10> to accept the default addresses.
10. The message: *Device programmed and verified* will be displayed.

TUTORIAL 3.2:

Programming an EPROM or Microcontroller from a file on disk

1. Type EPROM <ENTER>

2. Select the part you are going to programme using PART/TYPE. Select the device type you are wishing to use and press <ENTER>. You will then be given a list of manufacturers who support the selected device type. Select the appropriate manufacturer and press <ENTER>. You will then be prompted for the part name. Choose the correct part name from the menu you are given and press <ENTER> to accept. The details of the device you have chosen will now be displayed at the bottom of the screen.
3. Ensure that the buffer size is large enough to hold the data. You can change the buffer size using BUFFER/SIZE. See section 6.2.1 for more information.
4. Select FILE/LOAD
5. Specify the path and filename where your data is located.
6. Select the file format used (HEX auto-recognition will detect any of the HEX formats available)
7. Press <F10> to accept the default settings and load the data. A checksum will be displayed on the screen.
8. Place the device to be programmed in the programmer.
9. Select ACTION/AUTOPROGRAMME. Press <F10> to accept the default addresses.
10. The message: *Device programmed and verified* will be displayed.

TUTORIAL 3.3:

Programming a PAL or GAL

1. Type PAL <ENTER>.
2. Select the part you are going to programme using PART/MANUFACTURER. Select the manufacturer and press <ENTER>. Choose the device part name from the list on the screen by highlighting the correct part and pressing <ENTER>.
3. Select FILE/LOAD
4. Specify the path and filename where your JEDEC file is located.
5. Press <F10> to accept the defaults. A checksum will be displayed once the file has loaded.
6. Place the device to be programmed in the ZIF socket.
7. Select ACTION/AUTOPROGRAMME. The device will now be erased (where relevant), programmed, verified and any test vectors present in the JEDEC file will be applied.
8. The message: *Device programmed and verified* will be displayed.

4. Using the Programmer Software

4.1 Basic Software Configuration

The operating software contains the following programmes.

EPROM.EXE : Used to programme the various Memory and Microcontroller device types.

PAL.EXE : Used for Programmable Logic.

CHIPTEST.EXE: Used to identify and test TTL and CMOS logic devices.

These are menu driven programs which contain all the features required to programme and test devices. The actual programming information for individual devices is held in library files, which allows easy updating of the software to accommodate new devices.

In order for these programmes to run correctly, the appropriate .INI files must be configured correctly. If the software has been installed as described in section 1.3, the default .INI files will contain the correct path and directory information required by each of the programmes. If however, on running EPROM for instance, the message:

Cannot find database. Modify the ini file

is displayed, then the [Directory] section of the IT-EPROM.INI file is not pointing to the correct path. If either the PAL or Memory libraries are not correctly mapped, after selection of a part, the first selection off the ACTION menu will display the message:

Bad or missing library file

Using any ASCII editor, edit the IT_EPROM.INI and IT_PAL.INI files. If the software has been installed on C:\ICETECH, the [Directory] section of the .INI file should read:

```
[Directory]
Memory Libraries=C:\ICETECH\Libs\Memory
Pal Libraries=C:\ICETECH\Libs\Pals
Pars=C:\ICETECH\Pars
```

Batch software is also available which allows the programmer to be operated from DOS batch files. Details are available from ICE Technology.

For some programmable logic devices, separate programmes are supplied to handle non-JEDEC standard files.

4.2 Screen Layout

Each programme displays information as follows :

Menu headings are displayed across the top of the screen
Device programming details are displayed in the bottom window.
Buffer contents are displayed in the main window.

4.3 General Menu Selection

The software has been designed to make the programmer as simple as possible to use. All functions are contained within the menu system. The menus are obtained by pressing the <ESCAPE> key. The user can then move around the menus by pressing the cursor keys. To select an option press <ENTER>.

Each menu function is available by pressing the first letter of that menu. For instance, the PART pull-down menu is available by pressing the 'P' key. Some options can be selected by pressing ALT+letter. These are indicated by a triangle next to a letter on the menu display. For example, Auto-Programme can be selected by pressing ALT+A. A full list of shorthand keystrokes is given in Appendix A.

4.4 Dialogue Boxes

The menu system makes extensive use of dialogue boxes. Within these boxes the following prompt types may appear :

Prompt **:** requires you to type in some information
> will display a series of options when you press <ENTER>.

Inside the dialogue boxes use the cursor keys to move between options; use <ENTER> to select the contents of the dialogue box. Use <F10> to accept all chosen options within the window. If you wish to abort from the window, press <ESCAPE>.

4.5 Changing the Settings

Before starting, you may wish to change any of the following. They are found in the OTHER menu. This will have to be done for both EPROM and PAL :

OTHER/PARALLEL PORT NO
Selects parallel port to be used. This auto-detects each time that the software is accessed when the port setting in the INI file does not agree with the actual port setting. The setting is then stored in the appropriate INI file on exit. Once the parallel port has been chosen it will be remembered when next using the programmer.

OTHER/READ COLOUR INFO
To choose whether the display is MONO or COLOUR
OTHER/SNOW PRECAUTIONS
Usually set to OFF but set it to ON if you have problems with snow on your monitor. (affects older PC's with CGA)

OTHER/HI-RES MODE
Select ON for 40/50 line mode. Effective only with EGA and VGA monitors.

4.6 Using a mouse

All three application programmes can be mouse driven. However your mouse driver software must previously have been installed. This is normally done automatically in your CONFIG.SYS file (by loading a device driver such as MOUSE.SYS) or in your AUTOEXEC.BAT file (by invoking a mouse driver

programme such as MOUSE.COM). If you are unsure how to do this please refer to your system manual or the manual supplied with your mouse. **EPROM.EXE, PAL.EXE** or **CHIPTEST.EXE** automatically enable the mouse if one is installed (see later if you wish to disable it).

4.6.1 Left Hand Mouse Button (LMB)

The LMB is used to select items in a list, such as a menu selection or a device name from the device list. Simply move the mouse to the required item and press the LMB. If the item is a menu heading, such as FILE, ACTION etc. then this will open up that particular menu. If the item is a sub-menu or part of a list this will highlight the chosen item. To select the highlighted item DOUBLE CLICK on the LMB. i.e. QUICKLY press and release the LMB twice. DOUBLE CLICKING performs the same action as PRESSING <CR>

4.6.2 Right Hand Mouse Button (RMB)

Pressing the RMB is exactly the same as pressing **<ESC>.**

4.6.3 Middle Mouse Button (MMB)

Pressing the MMB is exactly the same as pressing **<F10>**. When this key is a possible input the window being displayed will have F10 in the bottom border. Pointing at the text F10 and pressing the LMB will also produce the same result (For two button mouse users.)

4.6.4 Disabling the Mouse

To disable the use of the mouse invoke each of the programmes with the /NM option. e.g.

```
EPROM /NM
```

4.7 Edit and View modes and Scroll Bars

In order to observe data in the buffer without risking modification, a VIEW mode is available in the BUFFER menu. This will only allow scrolling, paging etc. Mouse users can enter this mode directly by pointing at the buffer window and pressing the LMB. In order to make moving around the buffer much easier SCROLL BARS have been added to the window borders.

4.8 Running the Software under Windows

All three programmes can be run in a DOS window under WINDOWS 3.1 or Windows 95. Example .PIF files and .ICO files are included. Use optional parameters if required. The mouse functions should work in a DOS window in 386 enhanced mode in WINDOWS 3.1 or Windows 95, but if you have trouble see section 10 of the WINDOWS readme file.

4.9 Using the Programmer Software over a Network

It is possible to run the programmer software from a network hard drive with the programmer itself connected to the local computer. However, the appropriate .INI files will have to be modified in order to map the PAL and Memory libraries, and PARS sub-directory to the correct full path name. The **EPROM.EXE** and **PAL.EXE** files use their respective .INI files to locate the Parts menu and device libraries.

If the PARS sub-directory is not correctly mapped, on running the software, the message:

```
Cannot find database. Modify the .ini file
```

will be displayed.

If either the PAL or Memory libraries are not correctly mapped, after selection of a part, the first selection off the ACTION menu will display the message:

Bad or missing library file

4.10 Using EMS Software

The programmer is supplied with EMS software which will enable expanded memory to be accessed if it is available. This requires an expanded memory manager compatible with EMS 3.2 standard. It allows up to 8 Mb of memory and will run on any PC with an expanded memory board or a 286 or 386 with an expanded memory manager. The software will not give the EMS option if a driver such as EMM386.EXE is not installed correctly.

If the EMS function is working correctly, it will be possible to increase the buffer size, for example, to 4096K without problem, providing there is enough memory available. If this is not possible, and the maximum it can be set to is 384K for instance, then the EMM386 driver has not been correctly installed. Check the CONFIG.SYS file in the root directory on your hard drive to check that the following line exists:

```
DEVICE=C:\DOS\EMM386.EXE
```

Ensure that the **NOEMS** option has **not** been included on this line. DOS maybe replaced by WINDOWS if you are running under Windows 95.

4.11 Saving software settings

When quitting the software, settings such as the selected part and the last loaded filename are automatically saved in the appropriate .INI file. An environment option is also available which allows the full device configuration of any number of devices to be stored and re-loaded easily and quickly. This is explained in detail in section 5.2.

5. EPROM Software

This section covers all the Memory and Microcontroller device types which are supported using **EPROM.EXE**.

To use this programme, at the DOS prompt, type

EPROM <CR>

5.1 Part Selection

In order to tell the software which programming algorithm to use, it is necessary first to select the part which is being used.

5.1.1 To select a part

To do this select:

PART/TYPE

You will then be given a choice of device types to select. Choose between:

EPROMS
EEPROMS/FLASH/NVRAM
SERIAL(E)EPROMS
MICROCONTROLLERS
BPROMs

Highlight the type required or press the first letter, and press <ENTER>.

A list of manufacturers will then be displayed on the screen who support the selected device type. Use the up and down cursor keys to find the manufacturer of the part you are using. You can also press the initial letter of the manufacturer you need, e.g. pressing M will take the cursor down to MACRONIX. When the cursor is highlighting the required manufacturer, press <ENTER>.
The software then displays a list of all the supported parts of the manufacturer and type selected. Highlight the required part name and press <ENTER> again.
The details of the selected part will then be displayed in the bottom part of the screen, with the library version displayed at the top of the screen, just below the menu bar.

5.1.2 Changing the selected part

PART/PARTNAME

Once a device has been selected, this selects another part of the same manufacturer and type

PART/MANUFACTURER

Selects another manufacturer of the selected part type

5.1.3 Which part?

Generally it is necessary to select the part name which is as close as possible to that stamped on the chip you are using. For example, a PIC16C54 microcontroller in windowed versions might be marked PIC16C54, whereas an OTP version could be marked PIC16C54-XT. The software uses the correct

nomenclature according to the manufacturer's datasheet. Speed and package indicators are only required when they affect programming algorithms.

5.1.4 Automatic part selection

Within **EPROM.EXE** it is possible automatically to select a part using `PART/QUERY EPROM.` This will also identify unknown EPROMs, as long as they contain an electronic signature of a known device. See section 5.4.3 for further details.

5.1.5 Low Voltage Parts

Low voltage parts are selected exactly as other parts. In the device information in the bottom window, you will notice that the programme and verify voltages will be a combination of 5V, 3.3V and/or 1.8V. As most programmers are still unable to provide 1.8V and 3.3V circuits, manufacturers still allow the devices to be programmed at 5V. The LV40 can not only verify a low voltage device at its nominal operating voltage, it can also programme with voltages down to 1.8V as well.

5.2 Loading & Saving Files

Files can be loaded from disk into the data buffer and saved to disk after editing using the `LOAD` and `SAVE` options in the `FILE` menu. LOAD ENVIRONMENT will also load a file, along with the stored device parameters and programming configuration. See section 5.2.5.

5.2.1 Setting the Buffer Size

It is necessary to ensure that the buffer has been set to a size large enough to accept the contents of the file to be loaded. Do this using

`BUFFER/SIZE`

Enter the required size in Kbytes. The maximum buffer size available is determined by the available memory of the PC. The programmer software uses approximately 150K plus the chosen buffer size. The EMS software will use expanded memory for the buffer before using conventional DOS memory. If the buffer cannot be increased sufficiently see section 4.10 for information on installing EMS.

Some memory devices have configuration and protection capabilities. To allow for this additional information to be stored, the software gives the option of increasing the buffer in size to allow for a config byte. This byte is used by the environment option. See section 5.2.5.

5.2.2 Loading A File

Select:

`FILE/LOAD`

This then displays a number of different options:

`File Name`:
Enter the required filename (wildcards are allowed - see section 5.2.8)

`File Type:` Select the required file format (see section 5.2.3)

`File Window Low Address:`

Enter the required file start address Default = 0

`File Window High Address:`

Enter the required file end address Default = FFFFFFFF

`File Increment:`

Select from Every Byte, Every 2nd Byte etc. upto Every 4th Word

`Buffer Increment:`

Select from Every Byte/Word etc. upto Every 8th Byte/Word

`Buffer Start Address:`

CR Enter the required address. Default = 0

`Pre-Fill Buffer with FF:`

Select ON or OFF. ON will destroy all current buffer contents.

Pressing `<F10>` accepts the parameters and loads the selected file, displaying the checksum when loaded (See 5.2.7).

5.2.3 File Format

The file type can be selected by pressing return to see the options available. Choose from:

```
RAW/BINARY
ASCII-SPACE-HEX
ASCII - OCTAL
HEX - AUTO RECOGNITION
HEX - MOTOROLA S1, S2 OR S3 RECORDS
HEX - INTEL MCS86
HEX - TECTRONIX
HEX - EXTENDED TECTRONIX
HEX - TI TAG
```

Choosing `HEX-AUTO RECOGNITION` is generally the easiest way of loading a HEX formatted file. However some linkers put additional headers at the start of the HEX file which can confuse the loader when using this option. Hence the other options.

`ASCII-SPACE-HEX` embodies the formats of `ASCII` space `HEX`, `ASCII ‘ HEX` etc. The software assumes that the file consists of HEX digits separated by non-HEX digits. Checksum information is not looked for and no separators are needed.

e.g.. If a file consisted of A BCD EF this would appear as

Address	0	1	2	3
Data	0A	BC	0D	EF

NOTE: USERS OF MICROCHIP PIC MICROCONTROLLERS:
When using the PICALC assembler to produce files for Microchip PICs, use the option `-F INHX8M` to save the file as an 8-bit Intel-Hex file, then load as `HEX-Auto Recognition`. **WARNING:** Any line within your code which sets the format to anything else will NOT be overridden by entering the INHX command at the command line.

5.2.4 Saving a file to disk

Files edited in the buffer or read from a device can be saved to disk in a number of different formats. It is recommended that frequently used files are saved in binary mode to save disk space and reduce loading time.

FILE/SAVE

Displays two options:

Save All Device Areas: Saves the whole of the device address range, and where applicable, the config byte at the end of the buffer.

Save Memory Block: Saves a User defined address range from the buffer.

Both options allow the file to be saved in a number of formats:

RAW/BINARY
ASCII-SPACE-HEX
ASCII - OCTAL
HEX - MOTOROLA (S2 format)
HEX - INTEL

5.2.5 Saving Configuration

Should you use a number of different parts, all with different configurations, or even the same part requiring different programming options, it is possible to save, and subsequently re-load, the whole programmer environment, including filename, programming options, device start and end address etc.

FILE/SAVE ENVIRONMENT

This stores the whole of the current set up in an Environment file. Simply type the required filename and press <F10>.

If you are using microcontrollers, or memory devices with configuration options, all the configuration bits can be saved in the enviroment file as an extra byte at the end of the file, or at a location specified in the manufacturers programming algorithm. This means that the next time the device is used, all of the configuration options have been configured exactly as they were the previous time, saving time, and reducing the chance of operator error.

To implement this feature, choose your required device and filename, and select ACTION/AUTOPROGRAMME. All the programming options are then displayed. Select all of your required set up options and press <F10>. Even with no device in the socket, the buffer will be modified with the appropriate configuration byte. On a National Semiconductor COP878x device, this is stored in location #1000, with and AMD 29F040 Flash PROM, this would be stored at address #80000.

Once the environment has been loaded again, the software will check this byte in order to set up the programming parameters automatically

5.2.6 Handling large files and devices

In order to facilitate the programming of EPROMs and the loading of files whose size is greater than the maximum buffer size the FILE/LOAD command (ALT-L) has the following parameters:

FILE WINDOW LOW ADDRESS (FWLA)
FILE WINDOW HIGH ADDRESS (FWHA)
FILE INCREMENT (FINC)
BUFFER START ADDRESS (BADD)

A byte is ONLY loaded into the buffer if its FILE ADDRESS is inside the window limits defined by the above (inclusive) range. Data at FWLA is loaded into the buffer at BUFFER START ADDRESS. The FILE INCREMENT works within this window.

If you have file loading problems please check the parameters above. Normally FWLA=0 and FWHA=FFFFFFFF (i.e. the whole file is to be loaded). However these parameters are stored each time the programme is quit. If in a previous session FWLA was changed to 10000 (HEX) then files with information in the range 0-FFFF will appear empty.

5.2.7 File Checksum

When a file is loaded into the buffer a checksum is calculated on the contents of the file and displayed on the screen.

This is displayed as three figures:

A) The sum of unsigned bytes with the carry being added to the MSB of the checksum.
B) The 2's complement of (a)
C) The 1's complement of (a)

NOTE: The checksum displayed on FILE/LOAD is the checksum of the FILE only, and does not necessarily include the whole address range of the selected device. A checksum of the device, or of a particular address range can be calculated using BUFFER/CHECKSUM as explained in 5.3.5.

5.2.8 Wild card Loading

Wild cards may be used in the file name to perform drive/directory searches. Wild cards consist of * and ?.

* matches to any number of characters
? matches to any one character

A list of matching file names is displayed alphabetically (Sub-Directories first) and the user may select using the cursor keys or the first letter of the name.

5.2.9 Viewing the contents of a file

It is possible to view the contents of a file without having to load it into the buffer. This enables you to check the contents before loading. In order to view a file, select:

FILE/VIEW

The first 100 lines (max. 74 characters per line) can be viewed by selecting VIEW FILE. Use ALT+X to view in HEX mode or ALT+I to view in ASCII mode. <ENTER> records the last filename chosen. Filenames or wild cards can be entered to select the file to be viewed.

5.3. Using the Editing Facilities

As well as loading from disk or reading a pre-programmed device, data to be programmed can be entered directly into the buffer from the keyboard. In addition, files can be amended using the BUFFER menu options**.**

5.3.1 Entering EDIT mode

To edit the buffer contents, select

BUFFER/EDIT

This allows you to type in HEX or OCTAL code or ASCII characters. Once EDIT is selected, the cursor will move to the HEX display in the buffer window. Use the TAB key to switch between HEX and ASCII. In HEX mode, only keys 0-9 and A-F are allowed. In OCTAL mode only keys 0 to 7 are allowed. The cursor keys can be used to move around the buffer.

5.3.2 Viewing without making changes

If you wish to view the buffer contents beyond the first screen without danger of making unintentional changes to the data, use

BUFFER/VIEW

This allows you to view the contents of the buffer as HEX or OCTAL code or ASCII characters. Once VIEW is selected, the cursor will move to the HEX display in the buffer window. Use the TAB key to switch between HEX and ASCII.

5.3.3 Quickly Moving Around the Buffer.

The <HOME> and <END> keys allow you to move more quickly. Press once to move to the beginning (end) of the line, twice to move to the beginning (end) of the page or three times to move to the beginning (end) of the file.

When you have finished editing or viewing the buffer, press <ESCAPE> to return to the main menu.

5.3.4 Quick buffer editing commands

BUFFER/FILL

Allows you to fill a defined area of the buffer with information in byte, word or long word format. Enter the buffer start and end addresses and the information to be inserted. Use less than 3 characters for a byte, 3 or 4 for a word, more than 4 for a long word. Press <F10> to accept.

BUFFER/COPY

Copies a defined block of buffer memory to a new location.

BUFFER/SEARCH

Searches for data in HEX, OCTAL or ASCII format. Use the Up/Down arrow in the dialogue box to move between HEX, OCTAL and ASCII data entry.

BUFFER/GOTO

Goes to a chosen address in the buffer.

BUFFER/SWAP BYTES

Swaps odd and even bytes within a defined range. Useful for 16 bit wide EPROMs.

5.3.5 Buffer checksum

The command:

BUFFER/CHECKSUM

Gives the option of performing a checksum on either the contents of a user defined range in the buffer, or the selected device.

Choosing MEMORY BLOCK CHECKSUM provides an option box: Buffer Start Address: default to 0
Buffer End Address:
default to end of buffer (+1 if config byte valid)
Checksum Type: Select between:

- Sum of Bytes
- Sum of Words
- Sum of Long Words
- CRC 32 bit

Choosing DEVICE CHECKSUM just gives the option of entering a device start and end address.

Whichever option is selected, the checksum is displayed as three figures:

A) The sum of unsigned bytes with the carry being added to the MSB of the checksum.
B) The 2's complement of (a)
C) The 1's complement of (a)

NOTE: The buffer could be larger than the size of the file which was loaded in. In order to obtain a checksum which correctly relates to the loaded file, specify the start and end address of the file. Checksums for microcontrollers are displayed as defined by the manufacturer's data sheet.

5.3.6 Buffer Display Modes

The contents of the buffer can be displayed in 8-bit HEX, 8-bit OCTAL, 12-bit (16-bit storage, LSB = low address) or 16-bit mode (LSB = low address). To change the display, select:

OTHER/DISPLAY MODE

and select the display mode required.

5.3.7 Printing the Buffer Contents

A print-out of all or part of the buffer contents can be obtained by selecting:

BUFFER/PRINT

Specify the start and end addresses of the block to be printed, and the name of the output device (e.g. LPT1)

5.4 Reading and Checking Devices

This section explains how to read the contents of a programmed chip, verify the contents against known data, check the device signature and identify a device from its signature, and perform bit tests and blank checks on devices.

5.4.1 Reading a Device

To read the contents of a device, select the correct part name from the PART menu as described in section 5.1 and place the device to be read in the programmer socket. If you do not know the part name, you can use PART/QUERY EPROM to identify the device in many cases.

Please note that some devices CANNOT be read, either because the chip type itself is designed not to be read for security purposes, or because the on-chip security features have been enabled when programming.

Before reading EPROMs, ensure that the buffer size is set large enough to contain the data (see section 5.2.1 on setting the buffer size).

NOTE: A 16-bit device with a maximum device address of 128K will require 256K bytes of buffer space.

To read the contents or part of the contents of a device, select

ACTION/READ

The programmer software then quickly accesses the library file for the programming algorithm required. If a microcontroller is the selected device, then the software automatically reads the whole device and places the data into the correct area of the buffer. With memory devices, an option box appears requesting Device Start and End address, and Buffer start address:

DEVICE START ADDRESS: Enter the first address from which you wish to start reading data from the device. Default = 0.

DEVICE END ADDRESS: Enter the last address from which you wish to read data in the device. Defaults to the highest address available for the selected device.

BUFFER START BYTE ADDRESS: Enter the buffer address where you wish to begin reading the data into. Default = 0.

Press <F10> to accept the options displayed and the device will then be read. If the selected device is a microcontroller, then the entire contents of the device are read and placed into the appropriate buffer locations. The contents of the device will then be visible in the buffer, and a checksum will be displayed.

Once the contents of a device have been read, they can be edited using the BUFFER/EDIT option, and then saved to disk, or programmed into another device. See section 5.3 for more detail on editing data.

5.4.2 Verifying the contents of a chip against a file.

To verify a chip's contents against known data, first select the part you are using. Ensure that the buffer size is large enough to handle the file. Then load in the file against which you need to verify the data. Place the device to be verified in the programmer and select:

ACTION/VERIFY

If the data matches, the message *VERIFY OK* will be displayed. If there is a mismatch, the message will indicate at what address it has failed to verify the data.

5.4.3 Verifying the device type and identifying unknown devices.

Many devices now contain Electronic Signature codes which enable the part to be identified electronically. It is possible both to verify a known device type against this ID code, and to identify unknown parts. The ID code is also used by your programmer to check that the correct device type has been selected before

proceeding with any actions that may damage a chip. However, not all devices contain this ID code, and the checks for electronic signature may themselves damage a chip without a signature, or one of a different number of pins from that expected. This means that these checks cannot be foolproof and need to be used with some caution.
To verify the signature of a device, choose

ACTION/VERIFY SIGNATURE

If the database contains a manufacturer's code and part code they can be verified against that of the device.

To identify an unknown device, select

PART/QUERY EPROM

The option box then requests the number of pins on the device: 24, 28, 32 or 40. Other sized parts cannot be identified electronically. The electronic signature of the device in the socket is then read, which is checked in the database to find out which device is in the programmer. If the device can be identified, the correct library is selected and its details are displayed at the bottom of the screen.

5.4.4 Checking if a device is blank

In order to programme a device, it must either be fully blank, or be capable of containing the information to be programmed in addition to the information it already contains. In order to test whether a chip can hold additional data, load the data to be programmed into the buffer (see section 5.2), and use

ACTION/BIT TEST

This checks to see if a non-blank device can be programmed with the current data in the buffer. For example, with an EPROM whose unprogrammed state is #FF, #F0 may be programmed to #00 but not to #0F.
In order to test whether a device is empty, select

ACTION/BLANK CHECK

This option checks if device is blank (all 1's or all 0's depending on the device).

NOTE: It is possible for a device to appear blank even though it is not. A secured device will often report Blank, as can a device which is not quite erased. If problems are encountered in programming a device, try erasing it for a longer period and then reprogramming.

Either the Blank Check or Bit Test can be performed when you select ACTION/AUTOPROGRAMME. This is selected from OTHERS/ PROGRAMMING OPTIONS. See section 5.5.1.

5.5 Programming Devices

IMPORTANT NOTE

ICE Technology only use the specified programming algorithms as provided by the device manufacturers. Whilst we make every effort to ensure that programming algorithms are correct, the said algorithms are not guaranteed by the manufacturers themselves and therefore cannot be guaranteed by ICE Technology.

Users are advised to ensure that devices work correctly in their systems before programming large quantities.

5.5.1 Programming Options

Prior to programming, there are a number of different options available, which can be selected from the

OTHERS/PROGRAMMING OPTIONS from the menu bar. These are generally more applicable when using the programmer for production quantity programming.

SERIAL NUMBERS: For production programming, it is often a requirement to have a unique serial number programmed into a designated location(s) on the selected device. The (AUTO)PROGRAMME option implements the option selected here. (See 6.5.2)

Select from:

```
None
Sequential Hex: LSB in low memory
Random Hex
Date, day, Time in HEX: YYYYMMDD WDHHMMSS
Sequential BCD: LSB in low memory
Random BCD
Date, day, Time in BCD: YYYYMMDD WDHHMMSS
```

SERIAL NUMBER Buffer Byte Address:

Enter the required start address of the serial number.

Serial Number Direction : Select LSB or MSB in Low memory. Selecting MSB will display the SN in the buffer in the same direction as in the options box

Serial Number Storage:

Select between NIBBLE, BYTE and WORD

SN Buffer Increment: Default is 1.

SERIAL NUMBER LENGTH:

Enter the number of bytes required for the serial number - upto 8 bytes.

SERIAL NUMBER (bytes 0-3):

Least significant 4 bytes (8 digits)

SERIAL NUMBER (bytes 4-7):

Most Significant 4 bytes (8 digits)

Auto device checksum: Select OFF, Before securing or After securing

If turned ON, after a device has been successfully programmed and verified, the device checksum is displayed, either before or after the Security fuse has been programmed.

Device Checksum Ignores SN: Select ON or OFF

If a unique serial number is being programmed into each device, the overall checksum will change with each program operation, invalidating the Checksum as a fail-safe checking procedure. If this option is turned ON, then checksum will be calculated using only the UNCHANGED data.

AUTO-PROGRAM PRE-TEST:

Select None, USE BIT TEST or USE BLANK CHECK.

This determines which kind of pre-testing is performed when the Auto-Program option is selected.

ASK FOR PARAMETERS ON REPEAT: Select ON or OFF

With this option set to ON, after each program operation the Device Programming Parameters box is displayed. However, if a large quantity of devices are being programmed with exactly the same information, it is often desirable not to display the parameters box. Selecting OFF also ensures protection against accidental alteration to the programming parameters.

FORCE MARGIN VERIFY: Select ON or OFF

After any program operation, an automatic verify pass is performed. This is at the nominal voltages specified by the manufacturer in the programming specification which may be a combination of 5V, 3.3V and/or 1.8V. Selecting ON provides additional verify passes at +/-5% of each of the designated verify voltages.

5.5.2 Programming Operation

Two different options are provided for programming devices.

ACTION/AUTOPROGRAMME

Performs either a bit test or blank check (as specified within the programming options) followed by a chip erase (for electrically erasable devices), then programme and verify, whereas

ACTION/PROGRAMME

simply programmes and verifies the device.
In normal circumstances, the AUTO-PROGRAMME option should be selected when you do not wish to retain any data which may already be in the device. Selecting:
ACTION/AUTOPROGRAMME

Displays a box which enables the required Programming Parameters to be selected. This information will differ depending on the type of device and the on-chip functions supported.

The most basic configuration will offer the following options:

DEVICE START ADDRESS: Enter the address where you wish the programming operation to start. Defaults to 0.
DEVICE END ADDRESS: Enter the last address of the device you need programming. Defaults to the device end address.
BUFFER START ADDRESS: Enter the address of the first byte in the buffer from where you want to start programming. Default is 0.

Should a Sequential Serial Numbers option have been selected in the Programming Options box (see 5.5.1), then the Serialisation parameters will be displayed here. These can also be modified if required.

SERIAL NUMBER (bytes 0-3): Least significant 4 bytes
SERIAL NUMBER (bytes 4-7): Most Significant 4 bytes

NOTE: The buffer will be updated from the parameters box with the NEXT serial number after each successful programme/verify cycle. The SN in the programme parameters box will always be the same as the SN in the buffer. The only exceptions to this are:

1. The 1st device will be programmed with the contents of the parameters box, NOT the buffer. The buffer is only updated AFTER programming
2. If the SN in the parameters box is changed manually between programme cycles, the device will be programmed with the data from the parameters box. Accidental alteration can be avoided by selecting OFF the Ask for Parameters on Repeat option in the Programming Options (see above).

Microcontrollers will generally offer a number of options in addition to the basic device parameters. See 5.6.

To accept the address selections or defaults, press <F10>.

The programmer will first check the device selection and position, then perform either a bit test or blank check, and then proceed with programming.

While a device is being programmed, a message will be displayed in the centre of the screen showing the address which has been reached. Following programming, an automatic verify will be conducted and if all is well, the message *DEVICE PROGRAMMED AND VERIFIED* will be displayed. If Auto Device Checksum has been selected, the device checksum will now be displayed.

If you need to programme another device, simply press <F10> to repeat the process. If ASK FOR PARAMETERS ON REPEAT has been set to ON, then the programming parameters box will be displayed, which allows you to change any of the options. Press <F10> to continue. If the repeat option is set to OFF, then the parameters box will not be displayed and a single key-stroke is all that is required to repeat the programming operation.

5.5.3 Overprogramming devices

If you have a device which already contains some data, and you wish to add data to it, use:
ACTION/PROGRAMME

The same options are provided as for AUTOPROGRAMME.

PROGRAMME will program a device with the selected contents of the buffer. Programme should only be used when programming part of a device while retaining data in other address locations. In normal circumstances, use AUTOPROGRAMME.

5.6 Microcontroller Programming

Microcontrollers generally have a number of specific features which make their programming different to that of memory devices. The LV40 is specifically designed to handle a wide range of microcontrollers, and can support most of the optional features which are available on them.

5.6.1 Address Locations

The programmer software handles address locations as they appear on the device. For example, default buffer and device start addresses are at the addresses where the data is held in the device. It is not possible to edit device start and end addresses of microcontrollers.

5.6.2 Security Features

For microcontrollers, the security features are supported as part of another option within the Programming Parameters box through the PROGRAMME and AUTOPROGRAMME functions. Security status is performed automatically before a Blank Check or Auto-Programme. To manually check the security status of a pre-programmed device, use:

ACTION/SECURITY STATUS

This gives the status of the device security features. The information displayed is representative of that on the device data sheet.

P represents a security bit which is programmed
U represents an unprogrammed security bit

IMPORTANT NOTE: Some devices cannot indicate security status when the security fuse(s) has been programmed. In these cases the device may look blank. Thus the device may fail programming attempts until it is erased. If any problems arise with devices that have on-chip security, please erase the device first and retry.

ACTION/VERIFY ENCRYPTION

If encryption is in effect/relevant this will verify the encrypted buffer data with the contents of the device.

5.6.3 Device - Specific Features

Where device specific features are supported, the options available will be displayed in the Programming Parameters box after selecting PROGRAMME or AUTOPROGRAMME. Microcontroller programming options include:

```
Security
Encryption Array
Oscillator type
Brown-Out protection
Watch Dog Timer
RAM size selection
```

Please refer to the manufacturers datasheets for more information on these options. If there are any programmable options which are not included, please contact ICE Technology for possible updates.

5.7 Erasing devices

For Flash EPROMs, use:
ACTION/ERASE

This electrically erases the data held on the chip.

To erase EEPROMs and NVRAMs which do not have a separate Erase command, fill the buffer with #FF's and programme the whole device.

6. Programmable Logic

This section covers the Programmable Logic device types which are programmed using **PAL.EXE**.

To use this programme, at the DOS prompt, type

PAL <CR>

6.1 Part Selection

In order to tell the programmer software which programming algorithm to use, it is necessary first to select the part which is being used.

6.1.1 To select a part

To do this, enter PAL.EXE and select

PART/MANUFACTURER

A list of supported manufacturers will be displayed on the screen. Use the up and down cursor keys to find the manufacturer of the part you are using. You can also press the initial letter of the manufacturer you need, e.g. pressing N will take the cursor down to NATIONAL SEMICONDUCTOR. When the cursor is highlighting the required manufacturer, press <ENTER>.

The software then displays a list of all the supported parts of the manufacturer. Highlight the required part name and press <ENTER> again.

The details of the selected part will then be displayed in the bottom part of the screen.
NOTE: Some devices which are supported on the programmer will not appear on the menu. This is due to the fact that they do not use JEDEC standard files and therefore are programmed using a separate piece of software. At present this affects Altera and Xilinx EPLDs. See section 6.10 and 6.11 for details of how to use this software.

6.1.2 Changing the part selected

Once a device has been selected, to select another part of the same manufacturer, select:

PART/PARTNAME

When a new part is selected which has a different fuse map from the current part, the fuse map displayed on the screen will alter.

6.1.3 Which part?

Generally it is necessary to select the part name which is as close as possible to that stamped on the device you are using. Selecting a different part, no matter how small the difference may seem, may result in the device being damaged Sometimes, however, a number of parts with different suffixes will come under the same part name in the software. The programmer will check the device signature and use the correct algorithm for the part you are programming. For example, AMD devices with a suffix /4 and /5 are both supported under the same part name. If in any doubt, contact ICE Technology.

When using an adapter to programme a non-DIL package, select the part as normal. Any pin differences are

taken account of by the adapter. Use ICE Technology adapters wherever possible, especially for devices where the non-DIL version is not a straightforward pin for pin conversion of the DIL version., usually devices with more than 40-pins.

6.1.4 Low Voltage Parts

Low voltage parts are selected exactly as other parts. In the device information in the bottom window, you will notice that the programme and verify voltages will be a combination of 5V, 3.3V and/or 1.8V. As most programmers are still unable to provide 1.8V and 3.3V circuits, manufacturers still allow the devices to be programmed at 5V. The LV40 can not only verify low voltage devices at their nominal operating voltage, it can also programme them with voltages down to 1.8V as well.

6.2 Loading & Saving Files

Files can be loaded from disk into the buffer and saved to disk after editing using the LOAD and SAVE options in the FILE menu.

6.2.1 File Formats

Standard JEDEC files can be loaded into the buffer. These can be produced by packages such as PALASM^{tm}, OPAL^{tm}, CUPL^{tm}, PLACE^{tm} etc. Contact device manufacturers or distributors for details of programmable logic design packages. Some packages are given free with ICE Technology programmers from time to time.

In addition to JEDEC files, Altera POF format files and HEX files for Xilinx devices are accepted by the separate programmes provided for these devices.

6.2.2 Loading a file

To load a file into the buffer, first ensure that the correct part has been chosen, then select

FILE/LOAD
You will then be prompted for the filename to load. The software will expect the file to be in the correct format for the part selected. If there is a mismatch a warning will be displayed. Once the file is loaded, the screen will show the total number of 1's in the file, followed by the checksum. The JEDEC file loaded will include any test vectors that are present. These will be applied automatically when the part is programmed.

6.2.3 Wild card Loading

Wild cards may be used in the file name to perform drive/directory searches. Wild cards consist of * and ?.

* * matches to any number of characters
* ? matches to any one character

A list of matching file names is displayed alphabetically (Sub-Directories first) and the user may select using the cursor keys or the first letter of the name.

6.2.4 Viewing the contents of a file

It is possible to view the contents of a file without having to load it into the buffer. This enables you to check the contents before loading. In order to view a file, select

FILE/VIEW

The first 100 lines (max. 74 characters per line) can be viewed by selecting VIEW FILE. Use ALT+X to view in HEX mode or ALT+I to view in ASCII mode. <ENTER> records the last filename chosen. Filenames or wild cards can be entered to select the file to be viewed.

6.2.5 Saving a file to disk

Files edited in the buffer can be saved to disk using;

FILE/SAVE

All current test vectors will be saved along with the fusemap. If a file has been converted from a PAL to a GAL using the -conv part, the next time it is loaded it will be in the correct format for the GAL part.

6.3 Reading and Checking Devices

This section explains how to read the contents of a programmed chip, verify the contents against known data, check the device signature and identify a device from its signature, and perform bit tests and blank checks on devices.

6.3.1 Reading a Device

To read the contents of a device, select the correct part name from the PART menu as described in section 6.1 and place the device to be read in the programmer socket.

Please note that some devices CANNOT be read, either because the chip type itself is designed not to be read for security purposes, or because the on-chip security features have been enabled when programming.

To read the contents or part of the contents of a device, select

ACTION/READ

The programmer software then quickly accesses the library file for the programming algorithm required. The contents of the device will then be visible in the buffer, and a checksum will be displayed.

Once the contents of a device have been read, they can be edited using the BUFFER/EDIT option, and then saved to disk, or programmed into another device. See section 6.7 for more detail on editing data.

6.3.2 Verifying the contents of a chip against a file

To verify the contents of a device against known data, first select the part type you are using. Load in the file against which you need to verify the data. Place the device to be verified in the programmer and select

ACTION/VERIFY

If the data matches, the message *VERIFY OK* will be displayed. If there is a mismatch, an error message will be displayed.

6.3.3 Checking if a device is blank

In order to test whether a device is empty, select:

ACTION/BLANK CHECK

This option checks if device is blank (all 1's or all 0's depending on the device).

NOTE: It is possible for a device to appear blank even though it is not. A secured device will often report Blank, as can a device which is not quite erased. If problems are encountered in programming a device, try erasing it for a longer period and then reprogramming.

6.4 Programming Devices

IMPORTANT NOTE

ICE Technology only use the specified programming algorithms as provided by the device manufacturers. Whilst we make every effort to ensure that programming algorithms are correct, the said algorithms are not guaranteed by the manufacturers themselves and therefore cannot be guaranteed by ICE Technology.

Users are advised to ensure that devices work correctly in their systems before programming large quantities.

6.4.1 Programming operation

Two different options are provided for programming devices.

ACTION/AUTOPROGRAMME

performs a chip erase (for electrically erasable devices), bit test, programme and verify, whereas

ACTION/PROGRAMME

simply programmes and verifies the device.
In normal circumstances when you do not wish to retain any data which may already be in the device, first select the part required, ensure that the data is correct in the buffer, and then choose

ACTION/AUTOPROGRAMME

Should the device have on-chip security, you will be prompted to select the Auto-Secure option, select either ON (all devices will be secured) or OFF (No security). The programmer will first check the device selection and position, then perform a bit test or blank check, followed by an Erase (if necessary, and possible!), then proceed with programming. While a device is being programmed, an activity bar will be displayed in the centre of the screen. Following programming, an automatic verify will be conducted and if all is well the message *DEVICE PROGRAMMED AND VERIFIED* will be displayed.

6.4.2 Overprogramming devices

If you have a device which already contains some data, and you wish to add data to it, use

ACTION/PROGRAMME

The same options are provided as for AUTOPROGRAMME. The PROGRAMME option programmes device with contents of buffer. Programme should only be used when programming part of the device while retaining data in other address locations. **In normal circumstances, use** AUTOPROGRAMME.

6.4.3 Single key programming

If you need to programme more than one device in exactly the same way, simply press <F10> at the end of programming to repeat the process. If Auto-Secure is selected to OFF, you are given the option to secure the device before proceeding again by pressing <F10>.

6.5 Erasing devices

For GAL, PALCE, PEEL and other electrically erasable devices, use

`ACTION/ERASE`

This electrically erases the data held on the chip.

6.6 Device Security Features

When a programmable logic device has been secured, it is usually **NOT** possible to check the security status. The device will generally read Blank. However, with devices whose Security status can be read, select

`ACTION/SECURITY STATUS`

To set the security on by default during all programming, select

`ACTION/AUTO-SECURE`

6.7 Editing the Buffer

Data to be programmed into programmable logic devices is usually created using a development package such as OPAL^{tm}, ABEL^{tm} PALASM^{tm}, etc. This takes your logic equations and translates them into a **JEDEC** file which contains the AND/OR array which will be programmed into the device. In general it will not be necessary to edit this AND/OR array, or fusemap. However, the programmer software gives the option to be able to do this, for example when entering simple designs without a logic compiler, or for educational purposes.

6.7.1 Editing the AND/OR Array

When a device is selected, the buffer is automatically adjusted to the correct size and layout for the chosen part. The AND array and OR array (if applicable) are highlighted separately on the display. Product lines are represented horizontally and OR lines vertically.

Relevant keys:

1 - enters a 1 in the fuse map
0 - enters a 0 in the fuse map

In AND array:
`ALT 1` : changes the whole horizontal product line to 1
`ALT 0` : changes the whole horizontal product line to 0
In OR array:
`ALT 1` : changes the whole vertical OR line to 1
`ALT 0` : changes the whole vertical OR line to 0

`BUFFER/BLANK FUSE MAP`

Sets all fuses to 0 and deletes the test vectors.

`BUFFER/SET FUSE MAP`

Sets fuses to 1 and deletes the test vectors.

6.7.2 Moving around the buffer

BUFFER/GOTO

Moves the cursor to the selected fuse number.

6.7.3 User's Electronic Signature (UES)

Selecting:
BUFFER/EDIT UES

Allows the user to enter the Users Electronic Signature (where relevant). This can be entered as either ASCII or HEX code and the number of bytes depends on the device in use. The UES can be used for stock control, labelling, revision numbers etc. The UES is completely separate from the main array, and in no way affects the operation of a device.

NOTE: Electrically and operationally equivalent devices do not necessarily have the same UES capabilities. For instance, an AMD PALCE22V10 has no UES. A Lattice GAL22V10 does. If your JEDEC file has been compiled for the AMD part, and you are using the Lattice part, a file incompatibility error will be displayed during a FILE/LOAD operation. In this instance, the error message can be ignored.

6.7.4 Viewing the Fusemap

To view the contents of the fusemap without making any changes, select:
BUFFER/VIEW

When viewing is complete press <ESC> to return

6.7.5 Buffer checksum

The command:

BUFFER/CHECKSUM

Performs a checksum on the contents of the buffer. This is displayed as two figures:

A) The number of '1's in the buffer.
B) simple 2-byte checksum

6.8 Converting PAL Files for GAL Devices

It is possible to use the programmer to convert old files which have been set up for non-eraseable PAL parts to work in GAL devices. To do this, select a -CONV part from the menu, load in the file, and select

BUFFER/CONVERT FUSE MAP

This will convert the fuse map from the -CONV part to the generic part. This is performed automatically if -CONV part is to be programmed. For example, if an existing design in a PAL16L8 is to be converted into a National Semiconductor GAL16V8 then do the following:

1. ALT+M (manufacturer)
2. Select NATIONAL SEMICONDUCTOR
3. Select GAL16V8-CONV from parts list
4. Select XPAL16L8 from conversion list
5. Load in the file using ALT+L

Then select CONVERT FUSE MAP in the BUFFER menu to perform the conversion. Alternatively choose ALT+A to convert and programme in one operation. Once a file has been converted it can be saved and used again as a file for the part to which it has been converted, without the need for any further action.

6.9 Test Vector Editing and Operation

Test vectors can be edited or created using the BUFFER/EDIT TEST VECTOR command. This will open a window with pin numbers and test vector numbers.
Relevant keys are:

0	Forces a logic zero onto pin
1	Forces a logic one onto pin
L	Expect a logic zero on pin
H	Expect a logic one on pin
C	(Clock). Apply all other signals with C=0 and then apply C=1 followed by C=0
X	don't care
Z	Expect high impedance
N	Power pin (Vcc or GND (Sometimes used as don't care but the test vector loader will issue a warning and convert all non power pins to X)
ALT+T	apply the test vectors.

Standard JEDEC Test Vectors can be loaded into the test vector buffer and applied to the chip through the VECTOR TEST (ALT+T) command. They are applied automatically when using Auto-Programme. Any errors are highlighted. Test vectors are applied to all 40 pins at the same time and any skew will be determined by the chip's capacitance. This small skew may cause problems on fast logic devices (<7ns) if asynchronous circuitry is implemented within the PLD.

6.10 Using .POF files for Altera MAX devices

Altera MAX devices do not use JEDEC format files and are therefore not implemented in PAL.EXE. Separate software enables the programmer to handle the .POF files which are used on these devices. If you do not have this software, please request it from ICE Technology or download it from the bulletin board.
The software contains drivers for each of the devices supported. If using an EPM7064 device for instance, type:

```
EPM7064 <ENTER>
```

This will list all of the options available for the EPM7064 device.

To programme a device, type in the part name and the file to be programmed;

```
EPM7064 file.pof -pN
```

where N is parallel port number (default is LPT1)

POF file editing is not implemented as the POF format data refers to ELECTRICAL ADDRESSES and DATA. Without recourse to the chip architecture data the information in the POF file is meaningless to the end user.

6.11 Programming Xilinx EPLDs

Xilinx EPLDs use HEX files rather than JEDEC files, and are not implemented in PAL.EXE. Separate software enables the programmer to handle the HEX files that are created by the Xilinx compiler. Each EPLD has a different .EXE file which operates in the following way. The example shown is for the XI7336.

All Xilinx software uses the following syntax.

```
XI7336 [FILENAME] [options]
```

FILENAME is the INTEL HEX file to be used. Only INTEL HEX files generated by the XEPLD development system can be used.

The following options are available :

-READ This can be used to read the Xilinx device in the socket and save the file to disk as an INTEL HEX. This is the default option and can be used in the following ways :

```
XI7336  readfile.hex
```

or

```
XI7336 readfile.hex -READ
```

-BLANK Useful for checking to see if a device is blank. No filename is required. The syntax is therefore :

```
XI7336 -BLANK
```

-VERIFY A verify will be automatically executed after programming but this option may be used to verify the device against an INTEL HEX file. The syntax is:

```
XI7336 vfyfile.hex -VERIFY
```

-PROG This is used to programme and verify a device from the INTEL HEX file specified. If the security has been set in the file then this will automatically be programmed. If not then a separate command with -SEC may be executed as shown below.

```
XI7336 prgfile.hex -PROG
```

-DISPSIG This allows the user signature to be displayed. This option does not depend on whether the security has been programmed and can be displayed in either case.

```
XI7336 -DISPSIG
```

-DISPSEC Allows you to read and display the security status of the device in the socket.

```
XI7336 -DISPSEC
```

-SEC This option allows you to programme the security bits on the Xilinx device. Once this has been done only the signature can be read from the device.

```
XI7336 -SEC
```

-Pn If the programmer is not on Parallel Port 1 then the port number can be set using this option. The default is 1. For example to read from port 2 you should type :

```
XI7336 testfile.hex -READ -P2
```

7. Using CHIPTEST.EXE

CHIPTEST.EXE is a utility which allows a whole range of ICs to be tested. These include 74 series and 4000 series logic devices, RAMS etc. To run the programme type :

```
CHIPTEST [P] [/NM] <CR>
```

where

`P` is the parallel port number (default = 1)

and

`/NM` disables the mouse

Functional tests can now be made on the more common 7400 and 4000 series devices as well as on static and dynamic rams. A search facility has also been included.

7.1 Testing a Device

To test a device :

1. Select the logic family or device to be tested
2. For logic devices select the device to be tested
3. Place the device in the ZIF socket and select `TEST DEVICE`
4. The test vectors for the device will now be applied. If an error occurs then the failed vector will be displayed along with was actually found when the device was tested. If no errors are found then a message will tell you that the test vectors were OK.

7.2 Identifying a Device

If you are unsure of a device then insert the device into ZIF socket. Select one of the logic families from the main menu (It doesn't matter which family is selected, both the 74 and 4000 vectors will be applied). Select `FIND A MATCH`. The search will last for about 5 seconds. If all the vectors for a particular part in the library match that of the device being tested then the device number is shown at the bottom of the screen. In some cases there may be more than one match. The software displays all of the devices whose tested vectors match that of the device under test.

7.3 Adding Devices to the User Library

Users can create test vectors to test less common parts. This is done in the same way as defined for PAL.EXE (see section 6.9). Ground and Vcc pins can be defined with the `G` and `V` descriptors within one test vector. Ground Pins are currently limited (in software) to PIN 20 of the ZIF socket. This will be extended at a later date.

To add a new device to chiptest, select

```
EDIT TEST VECTORS
```

and enter the test vectors required.

Then select

```
ADD TO USER LIBRARY
```

7.4 Chiptest Restrictions

The chip test software works by applying test vectors to the device to be tested. If the chip passes the tests the device is assumed good, otherwise it is assumed bad. The search mode works by applying test vectors for all devices in the data base and recording all devices that are good.

Herein lies the problem. The chip is only tested at a frequency determined by the rate at which test vectors can be applied to the device. With the programmer hardware this is typically 500 - 3000 test vectors per second (depending on PC). Although this is much faster than most portable chip testers, it is generally much slower than the target hardware. The chip may pass the test vectors but may fail in the target system. Similarly, output loading may be completely different between system and tester.

NOTE: If you suspect a chip is causing problems in your target system replace it. The only test we can guarantee is one with a negative result.

Testing of RAMs and LSI devices is slightly different. This is done by the microprocessor inside the tester interfacing to the chip directly. RAM tests are effectively performed on a 1MHz bus. These results should therefore be more reliable. However access times are not measured, hence please note the above warning.

7.5 CHIPTEST Example

The CHIPTEST software allows simple logic gates to be tested quickly and easily.

As an example suppose a batch of 74LS245 devices are suspect. To test the devices run the chiptest software by typing :

```
CHIPTEST N
```

where N is the parallel port where the programmer is connected (default is 1 if this parameter is missed out). The main screen now appears listing the types of devices that can be tested. In our case we select the 74 series family which is the first selection in the menu. This can be selected by moving the cursor to the correct line and pressing return or by double clicking on this selection with the mouse.

A second menu should now appear on the right hand side of the screen. Choose the SELECT DEVICE option. You will now be given the list of devices supported in the 74 series. Move to 74245 by pressing the cursor keys or by using the mouse to move up and down the scroll bar by the side of the window. Once this device has been selected then information you are ready to test the suspect devices. Place the device in the ZIF socket as shown below.

Select TEST DEVICE from the left hand menu. The test vectors will now be applied. If the device passes all of the test vectors then a message will be displayed on the screen informing you that the test vectors were OK. If the device fails a particular vector test then the failed vector is displayed on the screen. Press a key to return to the left menu. You could select FIND A MATCH to apply all of the test vectors to the device to see if the device matches with any in the library. The matches are listed at the bottom of the screen. To quit from the left hand menu press <ESC> or select the MAIN MENU option. This takes you back to the right hand menu where you can test a device from another family or choose QUIT to return to the operating system.

SECTION II

Using the
LV40 Portable
in Stand-Alone Mode

8. Getting Started

The LV40 is shipped ready for use. Internal memory and firmware contain all the software libraries and algorithms to enable the full range of memory, programmable logic and microcontroller devices to be programmed without the need of a PC. However, we recommend that users become equally familiar with the PC software to fully appreciate the operation of the programmer. This is described in Section I of this manual.

8.1 Latest Device Libraries & Firmware

If you have purchased your programmer through one of ICE Technology's distributors, it is advisable to contact ICE Technology to ensure that you have the latest firmware and software revisions. New revisions are released on a regular basis, and should the distributor hold stock for even a relatively short period of time, the software and library firmware may already have been superseded. Firmware is updated by software via the parallel or serial port of any IBM compatible PC - see section 10.4. Software is available 24-hours a day from our download service on our Web site.

Contact information is given in Appendix D.

8.2 Keypad Layout & Description

The 30-key multi-function keypad provides access to as many different options as possible with a minimum number of key presses. The basic layout comprises an ON/OFF/ESC button, a MENU button, a 16-digit HEX keypad, 0-F (white buttons), four Function keys, F1-F4, four cursor keys, a SHIFT ( ) key and an ENTER (↵) key. The Yellow A1 button and Blue A2 button give access to the additional ASCII characters marked in Yellow and Blue text.

F1 - F4

The Function keys, F1 - F4, are application specific, their function changing depending on which option has been selected. Their uses for each application are described in the appropriate sections of the manual. Also, when used in conjunction with the A1 and A2 keys, F1-F4 provide ASCII punctuation characters.

MENU

If any Application menu is displayed, the MENU key is a short-cut to all of the commonly used functions of that application. Use the Left/Right cursor keys to scroll through the short-cut functions, the displayed letter is the short-cut Hot-key.

It also quits from a function screen back to the application menu.

ESC

The ESC key, which is also used for ON/OFF, will generally cancel any operation and return the programmer to the Application menu.

Use the LEFT < > and RIGHT < > cursor keys for scrolling menu lists and highlighting menu options. Once an option has been selected using either the ↵key or the DOWN cursor, a sub-menu appears. The UP cursor will return the display to the application menu.

↵

The ↵or ENTER key, executes the selected option.

The , or SHIFT key, behaves very much as a Shift key on a PC. In CAPS mode, this key is used to select lower-case letters; In non-CAPS mode, this will select upper-case characters. It is also used for turning the unit OFF in conjunction with the ESC key.

HEX 0-F

Standard Hexadecimal keypad. Used for data entry in buffer edit (HEX) mode and PIN number entry for Protected Mode, and to select short-cut MENU options. As with F1-F4, when used in conjunction with the A1 and A2 keys, 0-F provide ASCII text and punctuation characters, as well as CAPS mode select.

A1 & A2

Pressing A1 will give access to all ASCII characters which are marked on the keypad in YELLOW text. When A1 has been pressed, a 1 will appear in the top right hand corner of the LCD display. This will disappear when any other key is pressed.

Pressing A2 will give access to all ASCII characters which are marked on the keypad in BLUE text. When A2 has been pressed, a 2 will appear in the top right hand corner of the LCD display. This will disappear when any other key is pressed.

8.3 Basic Operation

Turn the programmer ON by pressing:

<ON/OFF>

The display shows the MAIN Menu:

```
ICE Technology LV40 Portable X.XX-X
[F1]=EPROM [F2]=PAL [F3]=More [F4]=SETUP
```

X.XX-XX is the programmers internal firmware version number. This will typically read: 1.00-3

<F1> : initialises the EPROM firmware.
Required for programming memory and microcontroller devices

<F2> : initialises the PAL firmware.
Required for programming programmable logic devices

<F3> : will display more menu options:

[C]HIPTEST: for using the 74 & 4000 series logic IC tester
[E]MULATOR: for operating the on-board ROM emulator
(EMUL models only)

Press the first letter to select the option. These can also be selected from the main menu without having to go via <F3>.

<F4> : provides access to the SETUP menu:

Utilities : Run the Selftest diagnostic software, charge batteries or modify the LCD setup

Update : Allows the firmware or application libraries to be updated from PC software

Configuration :
Allows Protected Mode to be

enabled/disabled and Serial Port Baud rate
to be selected

All of these functions are described in detail in their appropriate section of the manual.

- To cancel an operation at any time, press:

<ESC>

- To return to the Main menu from any of the applications, select:

[Quit] then press <↔>

- Turn the programmer OFF whilst in any menu by pressing:

< > and <ON/OFF> at the same time
OR
<A1> followed by <ON/OFF>

9. QuickStart Tutorials

As with the PC mode, these QuickStart tutorials are provided to help you start using the programmer as quickly as possible. They should not be taken as a comprehensive guide to the programmer's capabilities. The following tutorials are provided:

TUTORIAL 9.1 Programming from a master device.

TUTORIAL 9.2 Programming from a file stored on disk

These tutorials are provided to get you using the programmer quickly for simple tasks. However, we recommend that you read the reference sections of the manual if you have any difficulty and before attempting any more complex operations.

TUTORIAL 9.1:

Programming from a master device.

Although the EPROM software is used for this example, the same steps apply for programming PAL devices.

Use the < > and < > keys for scrolling menus and highlighting options, and <↵> to select.

1. Turn the programmer ON by pressing <ON/OFF>
2. Select [EPROM] by pressing <F1>
3. Select the part you are using as the Master device by selecting [Part]/[Type]. Select the device type you are wishing to use and press <↵>. You will then be given a list of manufacturers who support the selected device type. Scroll through the list until the required manufacturer is displayed and press <↵>. You will then need to select the part. Choose the correct part name from the menu you are given and press <↵> to accept. The selected device code is displayed on the top line of the display
4. Insert the master device in the programmer
5. Select [ACTION]/[READ] to read the contents of the chip into the buffer. Press <↵> to step through the device and buffer addresses; default will read the whole device into the buffer from address 0. Press <F4> to accept. The device contents are read into the buffer memory. When the whole device has been read, a checksum is displayed
6. Take the Master out of the programmer.
7. Select the part you are copying to if it is different (see 2.)
8. Place the blank copy device in the programmer.
9. Select [ACTION[/[AutoProg]. Press <F4> to accept the default addresses.
10. The message: *Device programmed and verified* will be displayed.

TUTORIAL 9.2:

Programming from a file stored on disk via the parallel port

Although the PAL software is used for this example, the same steps apply for programming Memory or Microcontroller devices.

Use the < > and < > keys for scrolling menus and highlighting options, and <↵> to select.

1. Connect the LV40 to the PC parallel port via the cable provided and turn the programmer ON by

pressing <ON/OFF>

2. Select [PAL] by pressing <F1>
3. Select the part you are using by selecting [Part]/[Manufacturer]. You will be given a list of manufacturers who support programmable logic. Scroll through the list until the required manufacturer is displayed and press <↵. You will then need to select the part. Choose the correct part name from the menu you are given and press <↵ to accept. The selected device code is displayed on the top line of the display
4. Select [File]/[Load JEDEC via PC Port]
5. Using the PC download utility software, Specify the path and filename where your JEDEC file is located and press <ENTER>
6. A checksum will be displayed once the file has loaded.
7. Place the device to be programmed in the ZIF socket.
8. Select [Action/AutoProgramme]. The device will now be erased (where relevant), programmed, verified and any test vectors present in the JEDEC file will be applied.
10. The message: *Device programmed and verified* will be displayed.

10. Using the Programmer

The programmer operating system contains the following applications:

- **EPROM** : Used to programme memory and microcontroller devices.
- **PAL** : Used for all programmable logic devices.
- **CHIPTEST** : Used to identify and test TTL and CMOS logic devices
- **EMUL** : Used to emulate ROM devices (EMUL models only - see separate Emulator manual)

These applications are all accessed through the MAIN menu which is displayed at power on. The menu selections of each of the applications provide all of the features required to programme and test devices. All programming information is held in library files which can be easily updated from a PC parallel or serial port.

10.1 Menu Selection

Each menu has been organised for maximum ease of use. The MAIN menu, displayed at power on gives the following options:

```
ICE Technology LV40 Portable X.XX-X
[F1]=EPROM [F2]=PAL [F3]=More [F4]=SETUP
```

`X.XX-XX` is the programmers internal firmware version number. This will typically read: `1.00-3`

`<F1>` : initialises EPROM
`<F2>` : initialises PAL
`<F3>` : will display more menu options
`<F4>` : initialises the `SETUP` menu

In any of the application menus, the currently selected option is encased in `[ ]`, for example:

```
EPROM      SelectedDevice
[Buffer]  Action  Part  Quit
```

Pressing `<↔>` or `< >` will display the `Buffer` menu. The LEFT/RIGHT cursor keys (/) will move the select window to another option if required. Some menus, including the Buffer menu will contain more options than can be displayed at any one time, this is signified by cursor arrows at each end of the bottom row of the LCD:

```
EPROM Buffer   SelectedDevice
  [Edit] View Fill Copy Search
```

The LEFT/RIGHT cursor keys scroll the menu around, leaving the select window stationary. Once the required option is highlighted, press `<↔>`.

The `ESC` or `MENU` keys will return the display to the sub-menu, the `< >` will then return to the main application menu.

10.2 Dialogue Prompts

Wherever possible, the menu system will offer only a minimum number of options which can be accessed using just the Enter key and cursor keys. However, many functions within the ACTION menu require input from the user, for example Buffer and Device Start & End addresses, and device options. Where a defined set of device options are defined in the programming specification, then only these options will be available. Within the ACTION menus, the following prompt types may appear :

Prompt :requires the user to type in the appropriate information

will display a series of options in a menu, more than can be displayed on the screen.
Use LEFT/RIGHT cursors to select

Once the required option has been entered/selected, use <↵ to move onto the next prompt. Once all options have been selected, press <F4> to accept and proceed with the action. If you wish to abort from the parameters menu, press <ESC>.

10.3 Display Options

Before starting, you may wish to change the display options to suit your requirements. This is achieved by selecting:

<F4>SETUP

from the MAIN menu on start up.

In the SETUP menu, select [Utilities]. This displays:

```
SETUP Utilities
[Backlight] Contrast Charger Selftest
```

[Backlight]

If you have the optional backlit LCD fitted, this option will toggle the backlight state - that is, it will turn it ON/OFF depending on the current state

[Contrast]

This allows the contrast of the LCD to be changed to suit the user. Use the LEFT/RIGHT cursors to move the bar. Once the contrast is set correctly, press <↵ or <MENU> to return to the Setup Utilities menu.

The UP cursor or ESC key return to the SETUP menu.

10.4 Updating the Programmer

Both Libraries and programmer firmware are updated from a PC software utility, using either the parallel or Serial Port. If you are using the RS232 serial port at a baud rate other than 9600, this will need to be configured using:

[Configuration]/[RS232]

Select between 300 to 19200 Baud

From the SETUP menu on the programmer, select:

`[Update]/[Libraries]`

The programmer will then report *Send new Libraries...*

See Manual Addendum available from ICE Technology for full details on the LV40 update utility programme (see Appendix D).

10.5 Selftest

The LV40 is provided with selftesting diagnostics. This can either be run from the PC software, or from the internal firmware. The tests check all the hardware associated with programming devices and should be run if you encounter any programming problems. If Selftest reports any errors, contact ICE Technology for assistance. Please make a note of the selftest results in order to assist with Technical Support. See Appendix D for contact details.
To run the Selftest, select

`<F4>SETUP`

from the MAIN menu on start up.

In the SETUP menu, select:

`[Utilities]/[Selftest]`

The first part of the Selftest displays the programmer version number. Typically: `LV40-1.01`

`Press A to continue`

A firmware checksum is then calculated. An appropriate message is displayed: OK/BAD.

`Press B to continue`

The programmer RAM is tested. This may take a few seconds. Again OK is displayed if all is well.

`Press C to continue`

An internal check of pin resources is then performed. The @ symbol being displayed each time a test passes.

`Press D to continue`

Each of the pin drivers are then tested. This gives a very clear indication as to whether the programmer is OK for programming devices. If any of these tests fail, please make a note of the displayed text and press any key to continue.

NOTE: If any errors are reported during Selftest, contact ICE Technology BEFORE programming any devices.

10.6 Recharging the Batteries

The programmer can be used as a battery charger. In order to start recharging the batteries, the programmer needs to be set up in CHARGER mode and connected to the mains using the mains adapter provided.

To charge the batteries, select

<F4>SETUP

from the MAIN menu on start up.

In the SETUP menu, select:

[Utilities]/[Charger]

Then select the CHARGEMODE required:

[Slow] Fast Trickle

Slow is the default setting.

When the batteries are charging, the LCD display on the programmer will show the charge status:

Time : Displays the length of time that charging has been taking place
Temp : Displays the battery temperature
BatV : Displays the current battery voltage
ChargeV : Displays the current Charging Voltage
Current : Displays the current being drawn
Mode : Displays the selected mode (Slow/Fast/Trickle)

Charging will stop once the batteries have been fully charged. Once charging is complete, the display will clear.

WARNING : D O NOT ATTEMPT TO CHARGE NON-RECHARGEABLE BATTERIES

10.7 Saving the Settings

When switching OFF the programmer, ALL the settings are stored, including display options, device selected and buffer contents. The OFF key is disabled during Device selection to avoid conflicts.

10.8 Protected Mode

Should the programmer be required to be taken to a site, or used on a production line, by somebody not familiar with the unit, it can be setup by an Engineer so that all the settings, including device and buffer contents, are protected from accidental change. Once at site, devices can be programmed and verified, but not read (this would destroy the buffer contents), and the buffer contents cannot be edited.

To Enable the Protected Mode, enter the application you are wishing to use (PAL/EPROM/CHIPTEST/EMUL), select the required device, load a file or read a master device, select the required programming options, then quit to the MAIN menu. Then select:

<F4>SETUP/[Configuration]/[Pin]

Enter a 4-digit HEX (0-F) or ASCII (A-Z) code. This code must then be re-typed to confirm. Press ESC at anytime to cancel, or the LEFT cursor to delete the last character entered. Once confirmed, type <Y> to confirm The settings are then saved and the Protected Mode is enabled

To disable the protected mode, select:

[File]/[Pin]

and enter the 4-digit PIN.

Should the PIN be lost, or forgotten, a PC utility is available to reset the programmer. See the manual

addendum for full details - available from ICE Technology (see Appendix D).
.

11. EPROM Software

This section covers all the Memory and Microcontroller device types which are supported using **EPROM.**

To use EPROM, select

`<F1>EPROM`

from the MAIN menu on start up.

The message *`Initialising Database`* is displayed while the programmer initialises the EPROM firmware and configures itself using the stored settings, this takes a few seconds.

11.1 Part Selection

In order to tell the software which programming algorithm to use, it is necessary first to select the part which is being used.

11.1.1 To select a part

To do this select:

`[PART]/[TYPE]`

You will then be given a choice of device types to select. Choose between:

```
EPROMS
EEPROMS/FLASH/NVRAM
SERIAL(E)EPROMS
MICROCONTROLLERS
BPROMs
```

Highlight the type required and press <↵>

A list of manufacturers who support the selected device type will then be available. Use the `Left/Right` cursor keys to find the manufacturer of the part you are using. You can also press the initial letter of the manufacturer you need, e.g. pressing `M` will jump to `MACRONIX`. When the cursor is highlighting the required manufacturer, press <↵>.

The message *`Building Device List..`* is displayed while the selected manufacturer libraries are initialised.

A list of all the supported parts of the manufacturer and type selected is then available. Highlight the required part name and press <↵> `again.`

The library filename is then displayed on the screen while it is being initialised. Once complete, the software jumps back to the EPROM menu and the selected device code is displayed.

Information on the selected device can be obtained from:

`[Part]/[Info]`

This displays information on the selected device:

- Library Name
- Library Version Number

- Electronic Signature Codes (where applicable): MC = Manufacturer Code, DC = Device Code
- Number of Pins
- Memory size

The screen can only display a number of the options at any time, so automatically scrolls to give all the information. Press `ESC, MENU` or `ENTER` to return to the application menu.

11.1.2 Changing the selected part

`[PART]/[DEVICE]`

Once a device has been selected, this selects another part of the same manufacturer and type

`[PART]/[MANUFACTURER]`

Selects another manufacturer of the selected part type

11.1.3 Which part?

Generally it is necessary to select the part name which is as close as possible to that stamped on the chip you are using. For example, a PIC16C54 microcontroller in windowed versions might be marked PIC16C54, whereas an OTP version could be marked PIC16C54-XT. The software uses the correct nomenclature according to the manufacturer's datasheet. Speed and package indicators are only required when they affect programming algorithms.

11.1.4 Automatic part selection

Within EPROM it is possible to automatically select a part using `[PART]/[QueryEprom].` This will also identify unknown memory devices as long as they contain an electronic signature of a known device. See section 11.4.3 for further details.

11.1.5 Low Voltage Parts

Low voltage parts are selected exactly as other parts. As most programmers are still unable to provide 1.8V and 3.3V circuits, manufacturers still allow the devices to be programmed at 5V. The LV40 can not only verify a low voltage device at its nominal operating voltage, it can also programme with voltages down to 1.8V as well.

11.2 Loading & Saving files

Files can be down-loaded into the programmer buffer memory from disk on a PC, and uploaded back to disk using the PC utility software.

11.2.1 Buffer Size

The LV40 is provided with 4Mbit (512K by 8) of internal buffer memory as standard. This can be factory upgraded to 8Mbit. Should a device you are trying to read or programme exceed the buffer size, the message:

Bad buffer Star/End Address

will be displayed.

To programme or read a large device without increasing the buffer memory, see section 11.2.4

11.2.2 Loading Files from PC

To load a file from disk on a PC into the programmer buffer memory involves using the LV40 Exchange utility software, as well as the [File]/[Load] option on the programmer.

See Manual Addendum available from ICE Technology for full details on the LV40 Exchange utility programme (see Appendix D).

11.2.3 File Format

The file types that can be loaded into the buffer via a PC are:

```
RAW/BINARY
ASCII-SPACE-HEX
ASCII - OCTAL
HEX - AUTO RECOGNITION
HEX - MOTOROLA S1, S2 OR S3 RECORDS
HEX - INTEL MCS86
HEX - TECTRONIX
HEX - EXTENDED TECTRONIX
HEX - TI TAG
```

Choosing HEX-AUTO RECOGNITION is generally the easiest way of loading a HEX formatted file. However some linkers put additional headers at the start of the HEX file which can confuse the loader when using this option. Hence the other options.

NOTE: USERS OF MICROCHIP PIC MICROCONTROLLERS:
When using the PICALC assembler to produce files for Microchip PICs, use the option -F INHX8M to save the file as an 8-bit Intel-Hex file, then load as HEX-Auto Recognition. **WARNING:** Any line within your code which sets the format to anything else will NOT be overridden by entering the INHX command at the command line.

11.2.4 Handling large devices

In order to facilitate the reading and programming of EPROMs whose size is greater than the maximum buffer size, the [Action]/[Read] and [Action]/[Prog/AutoProg] commands have the option of selecting the device Start and End addresses, and the buffer start address.

Once the device Start and End addresses have been set within the range of the buffer memory, that area of the device can be read/programmed.

For example: If you are wishing to copy an SGS-Thomson 27C801 (8Mbit EPROM) device, and you only have 4Mbit of buffer memory, the procedure is:

1. Insert the Master device in the programmer socket
2. Read the Master device, addresses 0-7FFFF, into the buffer
3. Insert the blank Copy device
4. Programme this with the address range 0-7FFFF
5. Insert the Master device
6. Read the device, addresses 80000-FFFFF, into the buffer

7. Insert the Copy device
8. Programme this with the address range 80000-FFFFF

11.2.5 File Checksum

When a file is loaded into the buffer a checksum is calculated on the contents of the file and displayed on the screen.

This is displayed as three figures:

A) The sum of unsigned bytes with the carry being added to the MSB of the checksum.
B) The 2's complement of (a)
C) The 1's complement of (a)

NOTE: The checksum displayed on `[File]/[Load]` is the checksum of the FILE only, and does not necessarily include the whole address range of the selected device. A checksum of the device, or of a particular address range can be calculated using `[Buffer]/[Checksum]` as explained in 11.3.4.

11.2.6 Saving a File to Disk

To save a file from the programmer buffer memory to disk on a PC involves using the LV40 Exchange utility software, as well as the `[File]/[Save]` option on the programmer.

See Manual Addendum available from ICE Technology for full details on the LV40 Exchange utility programme (see Appendix D).

11.3 Using the Editing Facilities

As well as loading from disk or reading a pre-programmed device, data to be programmed can be entered directly into the buffer from the keypad. In addition, files can also be amended using the `BUFFER` menu options**.**

In `Buffer/Edit` & `Buffer/View` modes, the function keys, `F1-F4` are assigned special functions.

`<F1>` : Toggles the Display Mode, Default is 8-bit HEX. Other modes available are 8-bit OCTAL, 12-bit HEX and 16-bit HEX.
`<F2>` : Switches between ASCII and HEX screen (Not available in OCTAL or12-bit HEX modes)
`<F3>` : Allows you to enter an address, which will then be displayed

11.3.1 Entering EDIT mode

To edit the buffer contents, from the `EPROM` menu, select

`[Buffer]/[Edit]`

This allows you to type in HEX or OCTAL code or ASCII characters. Once EDIT is selected, the cursor will move to the HEX display in the buffer window. Use the `<F2>` key to switch between HEX and ASCII. In HEX mode, only keys 0-9 and A-F are allowed. In OCTAL mode only keys 0 to 7 are allowed. The cursor keys can be used to move around the buffer. Press `<MENU>` to exit, saving any changes.

11.3.2 Viewing without making changes

If you wish to view the buffer contents beyond the first screen without danger of making unintentional changes to the data, use

`[Buffer]/[View]`

This allows you to view the contents of the buffer as HEX or OCTAL code or ASCII characters. Once VIEW is selected, the cursor will move to the HEX display in the buffer window. Use `<F2>` key to switch between HEX and ASCII. Press `<MENU>` to exit.

11.3.3 Quick Buffer Editing Commands

`[Buffer]/[Fill]`

Allows you to fill a defined area of the buffer with information in byte, word or long word format. Enter the buffer start and end addresses and the information to be inserted. Use less than 3 characters for a byte, 3 or 4 for a word, more than 4 for a long word. Press `<F4>` to accept.

`[Buffer]/[Copy]`

Copies a defined block of buffer memory to a new location.

`[Buffer]/[Search]`

Searches for data in HEX, OCTAL or ASCII format.

`[Buffer]/[Goto]`

Goes to a chosen address in the buffer.

`[Buffer]/[Swap]`

Swaps odd and even bytes within a defined range. Useful for 16 bit wide EPROMs.

11.3.4 Buffer checksum

The selection:

`[Buffer]/[Checksum]`

Gives the option of performing a checksum on either the contents of a user defined range in the buffer, or the selected device.

Choosing `MEMORY BLOCK CHECKSUM` provides an option box: `Buffer Start Address`: default to 0
`Buffer End Address`:
default to end of buffer (+1 if config byte valid)

`Checksum Type`: Select between:

```
Sum of Bytes
Sum of Words
Sum of Long Words
CRC 32 bit
```

Choosing `DEVICE CHECKSUM` just gives the option of entering a device start and end address.

Whichever option is selected, the checksum is displayed as three figures:
A) The sum of unsigned bytes with the carry being added to the MSB of the checksum.

B) The 2's complement of (a)
C) The 1's complement of (a)

NOTE: The buffer could be larger than the size of the file which was loaded in. In order to obtain a checksum which correctly relates to the loaded file, specify the start and end address of the file. Checksums for microcontrollers are calculated and displayed as defined by the manufacturer's algorithm.

11.4 Reading and Checking Devices

This section explains how to read the contents of a programmed chip, verify the contents against known data, check the device signature and identify a device from its signature, and perform bit tests and blank checks on devices.

11.4.1 Reading a Device

To read the contents of a device, select the correct part name from the PART menu as described in section 11.1 and place the device to be read in the programmer socket. If you do not know the part name, you can use [Part]/[QueryEprom] to identify the device in many cases.

Please note that some devices CANNOT be read, either because the chip type itself is designed not to be read for security purposes, or because the on-chip security features have been enabled when programming.

Before reading EPROMs, ensure that the buffer size is large enough to contain the data (see section 11.2.1).

NOTE: A 16-bit device with a maximum device address of 128K will require 256K bytes of buffer space.

To read the contents or part of the contents of a device, select

[Action]/[Read]

If a microcontroller is the selected device, then the software automatically reads the whole device and places the data into the correct area of the buffer. With memory devices, an option screen appears requesting Device Start and End address, and Buffer start address:

DEVICE START ADDRESS: Enter the first address from which you wish to start reading data from the device. Default = 0.

DEVICE END ADDRESS: Enter the last address from which you wish to read data in the device. Defaults to the highest address available for the selected part.

BUFFER START ADDRESS: Enter the buffer address where you wish to begin reading the data into. Default = 0.

Press <F4> to accept the options selected and the device will then be read. A checksum will be displayed at the end of the read operation.

Once the contents of a device have been read, they can be edited using the BUFFER/EDIT option, and then programmed into another device. See section 11.3 for more detail on editing data.

11.4.2 Verifying Device Contents Against the Buffer

To verify a chip's contents against known data, first select the part you are using. Place the device to be verified in the programmer and select:

[Action]/[Verify]

If the data matches, the message *VERIFY OK* will be displayed. If there is a mismatch, the message will indicate at what address it has failed to verify the data.

11.4.3 Verifying the device type and identifying unknown devices.

Many devices now contain Electronic Signature codes which enable the part to be identified electronically. It is possible both to verify a known device type against this ID code, and to identify unknown parts. The ID code is also used by your programmer to check that the correct device type has been selected before proceeding with any actions that may damage a chip. However, not all devices contain an ID code, and the checks for electronic signature may themselves damage a chip without a signature, or one of a different number of pins from that expected. This means that these checks cannot be foolproof and need to be used with some caution.

To verify the signature of a device, choose:

[Ation]/[VfySig]

If the database contains a manufacturer's code and part code they can be verified against that of the device.

To identify an unknown device, select

[Part]/[QueryEprom]

Select the number of pins on the device: 24, 28, 32 or 40. Other sized parts cannot be identified electronically. The electronic signature of the device in the socket is then read, which is checked in the database to find out which device is in the programmer. If the device can be identified, the correct library is selected and its details are displayed in [PART]/[INFO].

11.4.4 Checking if a device is blank

In order to programme a device, it must either be fully blank, or be capable of containing the information to be programmed in addition to the information it already contains. In order to test whether a chip can hold additional data, use

[Action]/[BitTest]

This checks to see if a non-blank device can be programmed with the current data in the buffer. For example, with an EPROM whose unprogrammed state is #FF, #F0 may be programmed to #00 but not to #0F. Select the address range to be checked, defaults to whole device.

In order to test whether a device is empty, select:

[Action]/[BlankChk]

This option checks if device is blank (all 1's or all 0's depending on the device). Select device address range to be checked, defaults to whole device.

NOTE: It is possible for a device to appear blank even though it is not. A secured device will often report Blank, as can a device which is not quite erased. If problems are encountered in programming a device, try erasing it for a longer period and then reprogramming.

Either the Blank Check or Bit Test can be performed when you select ACTION/AUTOPROGRAMME.

11.5 Programming Devices

IMPORTANT NOTE

ICE Technology only use the specified programming algorithms as provided by the device manufacturers. Whilst we make every effort to ensure that programming algorithms are correct, the said algorithms are not guaranteed by the manufacturers themselves and therefore cannot be guaranteed by ICE Technology.

Users are advised to ensure that devices work correctly in their systems before programming large quantities and should incorporate further, regular functional tests into their Quality Assurance procedures.

11.5.1 Programming Options

Prior to programming, there are a number of different options available, which can be selected from the [Part]/[Options]menu.

Note: Serial Numbers are NOT available in Stand-Alone mode.

`Auto device checksum`: Select `OFF`, `ON-Before securing` or `ON-After securing`
If turned `ON`, after a device has been successfully programmed and verified, the device checksum is displayed, either before or after the Security fuse has been programmed.

`AUTO-PROGRAM PRE-TEST`:
Select `None`, `USE BIT TEST`, `USE BLANK CHECK` or `ERASE`.
This determines which kind of pre-testing is performed when the Auto-Program option is selected.

`FORCE MARGIN VERIFY`: Select `ON` or `OFF`
After any program operation, an automatic verify pass is performed. This is at the nominal voltages specified by the manufacturer in the programming specification which may be a combination of 5V, 3.3V and/or 1.8V. Selecting `ON` provides additional verify passes at +/-5% of each of the designated verify voltages.

11.5.2 Programming Operation

Two different options are provided for programming devices.

`[Action]/[AutoProg]`

Performs either a bit test and a blank check (as specified within `Part/Options`) followed by a chip erase (for electrically erasable devices), then programme and verify, whereas

`[Action]/[Prog]`

simply programmes and verifies the device.

In normal circumstances, the `AutoProgramme` option should be selected when you do not wish to retain any data which may already be in the device. Selecting:

`[Action]/[AutoProg]`

Displays a screen which enables the required programming parameters to be selected. This information will differ depending on the type of device and the on-chip functions supported.
The most basic configuration will offer the following options:

DEVICE START ADDRESS: Enter the address where you wish the programming operation to start. Defaults to 0.
DEVICE END ADDRESS: Enter the last address of the device you need programming. Defaults to the device end address.
BUFFER START ADDRESS: Enter the address of the first byte in the buffer from where you want to start programming. Default is 0.

Microcontrollers will generally offer a number of options in addition to the basic device parameters. See 11.6.

To accept the address selections or defaults, press <F4>.

The programmer will first check the device selection and position, then perform the selected pre-testing, and then proceed with programming.

While a device is being programmed, a progress bar is displayed on the screen. Following programming, an automatic verify will be conducted and if all is well, the message *DEVICE PROGRAMMED AND VERIFIED* will be displayed. If Auto Device Checksum has been selected in the Options, the device checksum will now be displayed.

If you need to programme another device, simply press <F4> to repeat the process, or any other key to return to the Action menu.

11.5.3 Overprogramming devices

If you have a device which already contains some data, and you wish to add data to it, use:

[Action]/[Prog]

The same options are provided as for AUTOPROGRAMME.

PROGRAMME will program a device with the selected contents of the buffer. PROGRAMME should only be used when programming part of a device while retaining data in other address locations. In normal circumstances, use AUTOPROGRAMME.

11.6 Microcontroller Programming

Microcontrollers generally have a number of specific features which make their programming different to that of memory devices. The LV40 is specifically designed to handle a wide range of microcontrollers, and can support most of the optional features which are available on them.

11.6.1 Address Locations

The programmer software handles address locations as they appear on the device. For example, default buffer and device start addresses are at the addresses where the data is held in the device. It is not possible to edit device start and end addresses of microcontrollers.

11.6.2 Security Features

For microcontrollers, the security features are supported as part of another selection within the Programming Parameters screen. Security status is performed automatically before a Blank Check or Auto-Programme. To manually check the security status of a pre-programmed device, use:

[Action]/[SecurityStatus]

This gives the status of the device security features. The information displayed is representative of that on the device data sheet.

P represents a security bit which is programmed
U represents an unprogrammed security bit

IMPORTANT NOTE: Some devices cannot indicate security status when the security fuse(s) has been programmed. In these cases the device may look blank. Thus the device may fail programming attempts until it is erased. If any problems arise with devices that have on-chip security, please erase the device first and retry.

[Action]/[VfyEncryption]

If encryption is in effect/relevant this will verify the encrypted buffer data with the contents of the device.

11.6.3 Device- Specific Features

Where device specific features are supported, the options available will be displayed in the Programming Parameters screen after selecting PROGRAMME or AUTOPROGRAMME. Microcontroller programming options include:

```
Security
Encryption Array
Oscillator type
Brown-Out protection
Watch Dog Timer
RAM size selection
```

Please refer to the manufacturers datasheets for more information on these options. If there are any programmable options which are not included, please contact ICE Technology for possible updates.

11.7 Erasing devices

For Flash EPROMs, use:
[Action]/[Erase]

This electrically erases the data held on the chip.

To erase EEPROMs and NVRAMs which do not have a separate Erase command, fill the buffer with #FF's and programme the whole device.

12. Programmable Logic

This section covers the Programmable Logic device types which are programmed using **PAL**.

To use PAL, select:

<F2>PAL

from the MAIN menu on start up.

The message *Initialising Database* is displayed while the programmer initialises the PAL

firmware and configures itself using the stored settings; this may take a few seconds.

12.1 Part Selection

In order to tell the programmer software which programming algorithm to use, it is necessary first to select the part which is being used.

12.1.1 To select a part

To do this select

`[Part]/[Manufacturer]`

A list of supported manufacturers can be accessed on the screen. Use the `Left/Right` cursor keys to find the manufacturer of the part you are using. You can also press the initial letter of the manufacturer you need, e.g. pressing N will take the cursor down to NATIONAL SEMICONDUCTOR. When the cursor is highlighting the required manufacturer, press <↵.

The message *`Building Device List...`* is displayed while the selected manufacturer libraries are initialised.
A list of all the supported parts of the selected manufacturer is then available. Select the required part name and press <↵ again.

Information on the selected device can be obtained from:

`[Part]/[Info]`

This displays information on the selected device:

- Library Name
- Library Version Number
- Number of Pins
- Number of Fuses
- Number of associated Test Vectors stored in the buffer

The screen can only display a number of the options at any time, so automatically scrolls to give all the information. Press `ESC, MENU or ENTER to return to the application menu.`

NOTE: Some devices which are supported on the programmer will not appear on the menu. This is due to the fact that they do not use JEDEC standard files and therefore are programmed using a separate piece of PC software. At present this affects only Altera and Xilinx EPLDs. See sections 6.10 and 6.11 for details of how to use this software.

12.1.2 Changing the part selected

Once a device has been selected, to select another part of the same manufacturer, select:

`[Part]/[Device]`

Select the required part from the given scroll list.

12.1.3 Which part?

Generally it is necessary to select the part name which is as close as possible to that stamped on the device you are using. Selecting a different part, no matter how small the difference may seem, may result in the

device being damaged. Sometimes, however, a number of parts with different suffixes will come under the same part name in the software. The programmer will check the device signature and use the correct algorithm for the part you are programming. If in any doubt, contact ICE Technology.

When using an adapter to programme a non-DIL package, select the part as normal. Any pin differences are taken account of by the adapter. Use ICE Technology adapters wherever possible, especially for devices where the non-DIL version is not a straightforward pin for pin conversion of the DIL version., usually devices with more than 40-pins.

12.1.4 Low Voltage Parts

Low voltage parts are selected exactly as other parts. The programme and verify voltages will be a combination of 5V, 3.3V and/or 1.8V. As most programmers are still unable to provide 1.8V and 3.3V circuits, manufacturers still allow the devices to be programmed at 5V. The LV40 can not only verify low voltage devices at their nominal operating voltage, it can also programme them with voltages down to 1.8V as well.

12.2 Loading & Saving Files

Files can be loaded from disk into the buffer via the parallel port and saved to disk after editing using the LOAD and SAVE utilities provided with the PC software.

12.2.1 File Formats

Standard JEDEC files can be loaded into the buffer. These can be produced by packages such as PALASM™, OPAL™, CUPL™, PLACE™ etc. Contact device manufacturers or distributors for details of programmable logic design packages. Some packages are given free with ICE Technology programmers from time to time.

12.2.2 Loading Files from PC

To load a file from disk on a PC into the programmer buffer memory involves using the LV40 Exchange utility software, as well as the [File]/[Load] option on the programmer.

See Manual Addendum available from ICE Technology for full details on the LV40 Exchange utility programme (see Appendix D).

12.2.3 Saving a File to Disk (1.1)

To save the programmer buffer contents to a file on disk on a PC involves using the LV40 Exchange utility software, as well as the [File]/[Save] option on the programmer.

See Manual Addendum available from ICE Technology for full details on the LV40 Exchange utility programme (see Appendix D).

12.3 Reading and Checking Devices

This section explains how to read the contents of a programmed chip, verify the contents against known data, check the device signature and identify a device from its signature, and perform bit tests and blank checks on devices.

12.3.1 Reading a Device

To read the contents of a device, select the correct part name from the PART menu as described in section 12.1 and place the device to be read in the programmer socket.

Please note that some devices CANNOT be read, either because the chip type itself is designed not to be read for security purposes, or because the on-chip security features have been enabled when programming.

To read the contents or part of the contents of a device, select

`[Action]/[Read]`

The programmer software then quickly accesses the library file for the programming algorithm required. The contents of the device will then be accessible via the `Buffer` menu and a checksum will be displayed.

Once the contents of a device have been read, they can be edited using the `BUFFER/EDIT` option, and then saved to disk, or programmed into another device. See section 12.7 for more details on editing data.

12.3.2 Verifying Device contents Against the Buffer

To verify the contents of a device against known data, first select the part type you are using. Place the device to be verified in the programmer and select:

`[Action]/[Verify]`

If the data matches, the message *`VERIFY OK`* will be displayed. If there is a mismatch, an error message will be displayed.

12.3.3 Checking if a device is blank

In order to test whether a device is empty, select:
`[Action]/[BlankChk]`

This option checks if device is blank (all 1's or all 0's depending on the device).

NOTE: It is possible for a device to appear blank even though it is not. A secured device will often report Blank, as can a device which is not quite erased. If problems are encountered in programming a device, try erasing it for a longer period and then reprogramming.

12.4 Programming Devices

IMPORTANT NOTE

ICE Technology only use the specified programming algorithms as provided by the device manufacturers. Whilst we make every effort to ensure that programming algorithms are correct, the said algorithms are not guaranteed by the manufacturers themselves and therefore cannot be guaranteed by ICE Technology.

Users are advised to ensure that devices work correctly in their systems before programming large quantities and incorporate further, regular functional tests into their Quality Assurance procedures.

12.4.1 Programming operation

Two different options are provided for programming devices:

[Action]/[AutoProg]

performs a chip erase (for electrically erasable devices), bit test, programme and verify, whereas

[Action]/[Programme]

simply programmes and verifies the device.
In normal circumstances when you do not wish to retain any data which may already be in the device, first select the part required, ensure that the data is correct in the buffer, and then choose

[Action]/[AutoProg]

Should the device have on-chip security, you will be prompted to select the Auto-Secure option, select either ON (all devices will be secured) or OFF (No security). The programmer will first check the device selection and position, then perform a bit test or blank check, followed by an Erase (if necessary, and possible!), then proceed with programming. While a device is being programmed, a progress bar is displayed. Following programming, an automatic verify will be conducted, followed by a Vector Test should any vectors be stored in the Buffer. If all is well the message *DEVICE PROGRAMMED AND VERIFIED* will be displayed.

12.4.2 Overprogramming devices

If you have a device which already contains some data, and you wish to add data to it, use

[Action]/[Programme]

The same options are provided as for AUTO-PROGRAMME. The PROGRAMME option programmes a device with the contents of the buffer. Programme should only be used when programming part of the device while retaining data in other address locations. In normal circumstances, use AUTOPROGRAMME.

12.5 Erasing devices

For GAL, PALCE, PEEL and other electrically erasable devices, use

[Action]/[Erase]

This electrically erases all the data held on the chip.

12.6 Device Security Features

When a programmable logic device has been secured, it is usually **NOT** possible to check the security status. The device will generally read Blank. However, with devices whose Security status can be read, select

[Action]/[SecurityStatus]

To set the security on by default during all programming, select

[Action]/[AutoSecure]

This will secure ALL devices without prompting each time to set the security.

12.7 Buffer Editing

Data to be programmed into programmable logic devices is usually created using a development package

such as OPAL™, ABEL™ PALASM™, etc. This takes your logic equations and translates them into a **JEDEC** file which contains the AND/OR array which will be programmed into the device. In general it will not be necessary to edit this AND/OR array, or fusemap. However, the programmer software gives the option to be able to do this, for example when entering simple designs without a logic compiler, or for educational purposes.

12.7.1 Editing the AND/OR Array

When a device is selected, the buffer is automatically adjusted to the correct size and layout for the chosen part. Product lines are represented horizontally and OR lines vertically. To edit the fuse map, select:

`[Buffer]/[EditFmap]`

The display will by default show fuse 0. Scroll through the fuse map using the cursor keys. The currently highlighted fuse number is constantly displayed.

Press `<F3>` to goto a required fuse address
Press `<F1>` to lock the fuse map into memory

Relevant keys:

1 - enters a 1 in the fuse map
0 - enters a 0 in the fuse map

Press `<MENU>` or `<ESC>` to quit the edit screen.

`[Buffer]/[BlankFmap]`

Sets all fuses in the fuse map to 0 and deletes the test vectors.

`[Buffer]/[SetFmap]`

Sets all fuses in the fuse map to 1 and deletes the test vectors.

12.7.2 Saving the Fusemap

When a file is loaded, or a device read, the fusemap is only stored in RAM. When the programmer is switched off, this data is lost. To store the fusemap in buffer memory, select:

`[Buffer]/[LockMemory]`

This saves the current fusemap. If the fusemap is subsequently edited, to save the changes, the `LockMemory` command must be selected, either from the Buffer menu or from within `Buffer/Edit`.
If any changes are made to the fusemap which you do not want to store:

`[Buffer]/[RefreshMemory]`

will overwrite the RAM fusemap with the data locked in memory.

12.7.3 User's Electronic Signature (UES)

Selecting:

`[Buffer]/[Edit UES]`

Allows the user to enter the Users Electronic Signature (where relevant). This is entered as either HEX code (default) or ASCII text (press `<F2>`). The length of the UES depends on the device in use. The UES

can be used for stock control, labelling, revision numbers etc. The UES is completely separate from the main array, and in no way affects the operation of a device.

<F1> locks the UES to memory

NOTE: Electrically and operationally equivalent devices do not necessarily have the same UES capabilities. For instance, an AMD PALCE22V10 has no UES. A Lattice GAL22V10 does. If your JEDEC file has been compiled for the AMD part, and you are using the Lattice part, a file incompatibility error will be displayed during a FILE/LOAD operation. In this instance, the error message can be ignored.

12.7.4 Viewing the Fusemap

To view the contents of the fusemap without making any changes, select:

[Buffer]/[View]

When viewing is complete press <ESC> or <MENU> to return

12.7.5 Buffer checksum

The command:

[Buffer]/[Checksum]

Performs a checksum on the contents of the buffer. This is displayed as two figures:

A) The number of '1's in the buffer.

B) A simple 2-byte checksum

12.8 Test Vector Editing and Operation

Standard JEDEC Test Vectors can be loaded into the test vector buffer along with the device .JED file, and can also be edited or created using:

[Buffer]/[Edit TV]

The display will show Vector 0, Pin 1. Enter the required vector pattern, using only the relevant keys. All other keys will be disabled.
Relevant keys are:

0	Forces a logic zero onto pin
1	Forces a logic one onto pin
L	Expect a logic zero on pin
H	Expect a logic one on pin
C	(Clock). Apply all other signals with C=0 and then apply C=1 followed by C=0
X	don't care
Z	Expect high impedance
N	Power pin (Vcc or GND (Sometimes used as don't care but the test vector loader will issue a warning and convert all non power pins to X)

<F4>

applies the test vectors from the edit screen. To apply from the Action menu, select:

[Action]/[VectorTest]

They are also applied automatically when using AutoProgramme. Any errors are highlighted. Test vectors are applied to all pins at the same time and any skew will be determined by the chip's capacitance. This small skew may cause problems on fast logic devices (<7ns) if asynchronous circuitry is implemented within the PLD.

13. CHIPTEST Software

CHIPTEST is a utility which allows a whole range of ICs to be tested. These include 74 series and 4000 series logic devices, RAMS etc. To run the programme select :

`<F3>[MoreOptions]/[ChipTest]`

or from the MAIN menu, just press `<C>`.

Functional tests can now be made on the more common 7400 and 4000 series devices as well as on static and dynamic rams. A search facility has also been included.

13.1 Testing a Device

To test a device :

1. Select the device type to be tested: `Logic/SRAM/DRAM`
2. Select the actual device to be tested - logic chips may be selected using the `FindMatch` function
3. Place the device in the ZIF socket and select `TEST DEVICE`
4. The test vectors for the device will now be applied. If an error occurs then the failed vector will be displayed along with was actually found when the device was tested. If no errors are found then a message will tell you that the test vectors were OK.

13.2 Identifying a Device

If you are unsure of a device then insert the device into ZIF socket. Select `[Logic]` from the Chiptest menu (this feature is not available for SRAM and DRAM parts). Both the 74 and 4000 vectors will be applied to the unknown device. Select `[FindMatch]`. The search will last for about 5 seconds. If all the vectors for a particular part in the library match that of the device being tested then the device number is displayed on the screen. In some cases there may be more than one match. The software displays all of the devices whose tested vectors match that of the device under test.

13.3 Chiptest Restrictions

The Chiptest software works by applying test vectors to the device to be tested. If the chip passes the tests the device is assumed good, otherwise it is assumed bad. The search mode works by applying test vectors for all devices in the data base and recording all devices that are good.

Herein lies the problem. The chip is only tested at a frequency determined by the rate at which test vectors can be applied to the device. With the programmer hardware this is typically 500 - 3000 test vectors per second. Although this is much faster than most portable chip testers, it is generally much slower than the target hardware. The chip may pass the test vectors but may fail in the target system. Similarly, output loading may be completely different between system and tester.

NOTE: If you suspect a chip is causing problems in your target system replace it. The only test we can guarantee is one with a negative result.

Testing of RAMs and LSI devices is slightly different. This is done by the microprocessor inside the tester interfacing to the chip directly. RAM tests are effectively performed on a 1MHz bus. These results should therefore be more reliable. However access times are not measured, hence please note the above warning.

13.4 CHIPTEST Example

The CHIPTEST software allows simple logic gates to be tested quickly and easily.
As an example suppose a batch of 74LS245 devices are suspect. To test the devices run the chiptest software by selecting :

`[Chiptest]/[Logic]`

This contains all of the 74 and 4000 series logic chip libraries. To select a specific device, use:

`[SelectDevice]`

You will now be given the list of devices supported in the 74 and 4000 series families. Move to 74245 by scrolling through the parts using the LEFT/RIGHT cursor keys. Place the device in the ZIF socket. Now select:

`[TestDevice]`

The test vectors will now be applied. If the device passes all of the test vectors then a message will be displayed on the screen informing you that the test vectors were OK. If the device fails a particular vector test then the failed vector is displayed on the screen. Press a key to return to the menu. You could select:

`[FindMatch]`

to apply all of the test vectors to the device to see if the device matches with any in the library. The matches are listed on the screen.

Appendix A: PC Software - Shorthand Keystrokes

For quick operation of ICE Technology Programmers the following shorthand keystrokes are available :-

	EPROM	**PAL**
Load	Alt+L	Alt+L
Save	Alt+S	Alt+S
View	Alt+I or Alt+X	Alt+I or Alt+X
Edit Buffer	Alt+E	Alt+E
Checksum	Alt+C	Alt+C
Fill Buffer		Alt+F
Goto Fuse		Alt+G
Goto Address		Alt+G
Edit UES	Alt+U	
Buffer Size	Alt+Z	
Read	Alt+R	Alt+R
Verify	Alt+V	Alt+V
Blank Check	Alt+B	Alt+B
OverProgramme	Alt+P	Alt+P
Autoprogramme	Alt+A	Alt+A
Query EPROM	Alt+Q	
Vector Test	Alt+T	
Manufacturer	Alt+M	Alt+M
Type	Alt+T	
Partname	Alt+N	Alt+N
DOS Command	Alt+D	Alt+D
Hi-Res Mode	Alt+H	Alt+H
Display Mode	lt+Y	

Appendix B: PC Software - Menu Options

This section gives a brief overview of each of the menu options available in EPROM.EXE and PAL.EXE and tells you in which section of the manual you will find more detailed information.

EPROM.EXE

Menu Option	Section
FILE MENU	
Load Load a binary or HEX formatted from disk and store it in the buffer.	5.2.2
Save Save an area of the buffer to a binary or formatted file on disk.	5.2.8
View File View the contents of a file in ASCII or Hexadecimal format.	5.2.9
Quit Exit the software and save the settings.	4.11
BUFFER MENU	
Edit Edit the contents of the data buffer.	5.3.1
View View the contents of the data buffer without making any changes.	5.3.2
Fill Fill an area of the buffer with a specified value.	5.3.4
Copy Copy an area of the buffer to another area.	5.3.4
Search Search an area of the buffer for a specified value or string.	5.3.3
Goto Goto a user specified area of the buffer.	5.3.3
Swap Bytes Swaps odd and even bytes. Useful for data into more than 1 EPROM.	5.3.4
Buffer Size Changes to size of the buffer.	5.2.1

Print 5.3.7
Prints out an area of the buffer

Checksum 5.3.5
Calculates a checksum over a given area of the buffer, or of the selected device.

ACTION MENU

Read 5.4.1
Read the device in the ZIF socket and store the data in the buffer.

Verify 5.4.2
Verify the data in the device against the data in the buffer.

Verify Signature 5.4.3
Checks the electronic signature of the device.

Bit Test 5.4.4
Checks to see if a non blank device can be programmed with the data in the buffer.

Blank Check 5.4.4
Checks to see if a device is blank.

Programme 5.5.2
Programmes data in the buffer into the device.(inc. Verify Sig & Verify)

Erase 5.7
Erases Electronically erasable devices.

Auto Programme 5.5.1
Same as Programme but with Bit Test or Blank-Check included.

Security Status 5.6.2
Checks the status of the device security bits.

Verify Encryption 5.6.2
Checks the Encryption status of a microcontroller

PART MENU

Type 5.1.1
Selects the type of device to be programmed.

Manufacturer 5.1.1
Selects the manufacturer of the device to be programmed.

Partname 5.1.1
Selects the partname of the device to be

programmed.

Query EPROM 5.4.3
Automatically select parameters for device in socket

OTHER MENU

Parallel Port 4.5
Tells the software which parallel port the programmer is connected.

Screen Colour 4.5
Selects either mono or colour displays.

Snow Precautions 4.5
Overcomes snow problems on old CGA displays.

Hi-Res Mode 4.5
Changes number of lines on EGA and VGA displays.

Display Mode 4.5
Switch between 8, 12, 16 bit display.

DOS Command
Enables a DOS command to be run from the software.

Programming Options 5.5.1
Selects user and device configuration options for programming

PAL.EXE

Menu Option Section

FILE MENU

Load 6.2
Load a binary or HEX formatted file from disk and store it in the buffer.

Save 6.2.5
Save an area of the buffer to a binary or formatted file on disk.

View File 6.2.4
View the contents of a file in ASCII or Hexadecimal format.

Quit 4.11
Exit the software and save the settings.
BUFFER MENU

Edit 6.7.1

Edit the contents of the data buffer.

View 6.7.4
View the contents of the data buffer without making changes.

Goto Fuse 6.7.2
Goto a defined fuse in the fuse map.

Blank Fuse Map 6.7.1
Sets all the fuses in the buffer to ‘0’.

Set Fuse Map 6.7.1
Sets all the fuses in the buffer to ‘1’.

Convert Fuse Map 6.8
Used to convert PALs to GALs.

Edit UES 767.3
Edits the Users Electronic Signature.

Edit Test vectors 6.9
Edit the test vectors for this file.

Checksum 6.7.5
Calculate checksum on the data in the fuse map.
ACTION MENU

Read 6.3.1
Read the device in the ZIF socket and store the data in the buffer.
Verify 6.3.2
Verify the data in the device against the data in the buffer.

Blank Check 6.3.3
Checks to see if a device is blank.

Programme 6.4.1
Programmes data in the buffer into the device.

Erase 6.5
Erases Electronically erasable devices.

Vector Test 6.9
Applies Test Vectors.

Auto Programme 6.4.1
Same as Programme but with bit test included.

Security Status 6.6
Checks the status of the device security bits.

Auto Secure 6.6

Automatically secures devices when programming

PART MENU

Manufacturer 6.1
Selects the manufacturer of the device to be programmed.

Partname 6.1
Selects the partname of the device to be programmed.

OTHER MENU

Parallel Port 4.5
Tells the software which parallel port the programmer is connected.

Screen Colour 4.5
Selects either mono or colour displays.

Snow Precautions 4.5
Overcomes snow problems on old CGA displays.

Hi-Res Mode 4.5
Changes number of lines on EGA and VGA displays.

DOS Command
Enables a DOS command to be run from the software.

Appendix C: Stand-Alone Mode - Menu Options

This section gives a brief overview of each of the menu options available in EPROM and PAL and tells you in which section of the manual you will find more detailed information.

EPROM

Menu Option	Section

BUFFER MENU

Edit 11.3.1
Edit the contents of the data buffer.

View 11.3.2
View the contents of the data buffer without making any changes.

Fill 11.3.3
Fill an area of the buffer with a specified value.

Copy 11.3.3
Copy an area of the buffer to another area.

Search 11.3.3
Search an area of the buffer for a specified value or string.

Goto 11.3.3
Goto a user specified area of the buffer.

Swap 11.3.3
Swaps odd and even bytes. Useful for programming data into more than 1 EPROM.

Checksum 11.3.4
Calculates a checksum over a given area of the buffer, or of the selected device.

ACTION MENU

Read 11.4.1
Read the device in the ZIF socket and store the data in the buffer.

Verify 11.4.2
Verify the data in the device against the data in the buffer.

Verify Signature 11.4.3
Checks the electronic signature of the device.

Bit Test 11.4.4
Checks to see if a non blank device can be

programmed with the data in the buffer.

Blank Check 11.4.4
Checks to see if a device is blank.

Programme 11.5.2
Programmes data in the buffer into the device (includes Verify Sig & Verify)

Erase 11.7
Erases Electronically erasable devices.
Auto Programme 11.5.2
Same as Programme but with Bit Test or Blank-Check included.

Security Status 11.6.2
Checks the status of the device security bits.

Verify Encryption 11.6.2
Checks the Encryption status of a microcontroller

PART MENU

Type 11.1.1
Selects the type of device to be programmed.

Manufacturer 11.1.1
Selects the manufacturer of the device to be programmed.

Partname 11.1.1
Selects the partname of the device to be programmed.

Info 11.1.1
Displays the device information

Query EPROM 11.4.3
Automatically select parameters for device in socket

Options 11.5.1
Selects the programme pre-test, verify and checksum options
PAL

Menu Option Section

BUFFER MENU

Edit 12.7.1
Edit the contents of the data buffer.

View 12.7.3
View the contents of the data buffer without making changes.

Goto Fuse 12.7.1
Goto a defined fuse in the fuse map.

Blank Fuse Map 12.7.1
Sets all the fuses in the buffer to ‘0’.

Set Fuse Map 12.7.1
Sets all the fuses in the buffer to ‘1’.

Edit UES 12.7.2
Edits the Users Electronic Signature.

Edit Test vectors 12.8
Edit the test vectors for this file.

Checksum 12.7.5
Calculate checksum on the data in the fuse map.

Lock Memory 12.7.2
Saves the current fusemap into buffer memory.
Refresh Memory 12.7.2
Loads the fusemap stored in buffer memory into RAM, overwriting current fusemap.

ACTION MENU

Read 12.3.1
Read the device in the ZIF socket and store the data in the buffer.

Verify 12.3.2
Verify the data in the device against the data in the buffer.

Blank Check 12.3.3
Checks to see if a device is blank.

Programme 12.4.1
Programmes data in the buffer into the device.

Erase 12.5
Erases Electronically erasable devices.

Vector Test 12.8
Applies Test Vectors.

Auto Programme 12.4.1
Same as Programme but with bit test included.

Security Status 12.6
Checks the status of the device security bits.
Auto Secure 12.6
Automatically secures devices when programming

PART MENU

Manufacturer 12.1.1
Selects the manufacturer of the device to be programmed.

Device 12.1.1
Selects the part name of the device to be programmed.

Info 12.1.1
Displays the device information

FILE MENU

SETUP

Menu Option Section

UTILITIES MENU

Backlight 10.3.1
Turns the backlight ON/OFF.

Contrast 10.3.1
Changes the LCD contrast setting

Charger 10.6
Selects the Battery Charger

Selftest 10.5
Runs the programmer selftest diagnostic software

UPDATE MENU 10.4
Use with the PC software utility - updates the programmer firmware & libraries

CONFIGURATION MENU

PIN 10.8
Enter the PIN number to enable/disable Protected Mode

Protected Mode 10.8
Enables the Protected Mode using the selected PIN and saves the settings

Appendix D: Technical Support & Contact Information

If you require technical support for any reason, please ensure that you have the following information to hand:

1. The Serial Number of your programmer
2. The Selftest results (see sections 2.3 and 10.5)
3. The Software/Firmware versions (reported in Selftest)
4. The Device type and library revision numbers

TECHNICAL SUPPORT CONTACT DETAILS:

U.K.

email: support@icetech.com

U.S.A.

email: icetechusa@icetech.com

TO DOWNLOAD SOFTWARE AND INFORMATION:

http://www.icetech.com

Guarantee Conditions

1. All programmers are guaranteed for a period of twelve months from the date of purchase, package adapters for 30 days.
2. This guarantee covers both parts and labour.
3. In the event of an item being found to be faulty, ICE Technology guarantees to repair or replace the item at its own discretion.
4. Repairs will be carried out on the premises of ICE Technology or its agents or distributors at the discretion of ICE Technology.
5. No item will be accepted by ICE Technology for repair or refund without an RMA number having been issued by ICE Technology.
6. All costs of return delivery to ICE Technology are the responsibility of the purchaser.
7. Return delivery costs of guaranteed items back to the customer will be met by ICE Technology where these are within the UK. The method of delivery will be wholly at the discretion of ICE Technology.
8. Return delivery outside the UK will be the responsibility of the purchaser and must be paid for before delivery will be made.
9. While every endeavor will be made to repair or replace any item as quickly as possible, no guarantees can be made on the time taken and ICE Technology cannot be held responsible for any loss or inconvenience caused.
10. ICE Technology cannot be held responsible for any loss or damage, consequential or otherwise, incurred while using its equipment.
11. While ICE Technology makes every endeavour to ensure that programming algorithms are correct, the said algorithms are not guaranteed by the manufacturers and therefore ICE Technology cannot make any guarantees regarding the algorithms implemented on its programmers. Users are advised to ensure that devices work correctly in their systems before programming large quantities.
12. All items except for custom made items or software are covered by a 30 day money back guarantee if not completely satisfied. Before returning any goods for refund, an RMA number must be obtained. ICE Technology reserves the right to make a handling charge .