Installing The Software

Check Your Software

ICE Technology’s WinLV software is shipped either on 3 HD diskettes or a CD. Make sure that you have been provided with all the disks required to install the software. Please either contact the ICE Technology sales office or download the latest software from our web site if you have bad or missing disks.

Latest Software

Due to ICE Technology’s commitment to continuously updating and adding new devices to our software, it is advisable to contact ICE Technology to ensure that you have been shipped the latest release of WinLV. New versions of the software are released periodically and so if you purchased your programmer from one of ICE Technology’s distributors it may have been held in stock for a while and so have an earlier version of WinLV. For information on latest software releases please contact ICE Technology sales office

Making A Backup

Before Installing the software it is advisable to make a back-up as soon as you receive the disks. For example, use:- COPY A:*.* B:

Running Setup in Windows 95/98 or Windows NT4.0

To install WinLV onto your PC:-

? Insert disk 1 into drive A

? Click on *Start* in Windows 95 or NT4.0 Desktop

? Select *Run…*

? Type “a:\setup” in the *Run* dialog box

? OR click on *Browse…* in the Run window to manually select the setup file on drive A

? Follow the on screen setup instructions

Connecting The Programmer

Computer Requirements

Make sure that your PC has the following minimum requirements otherwise you may experience problems using WinLV and the programmer:-

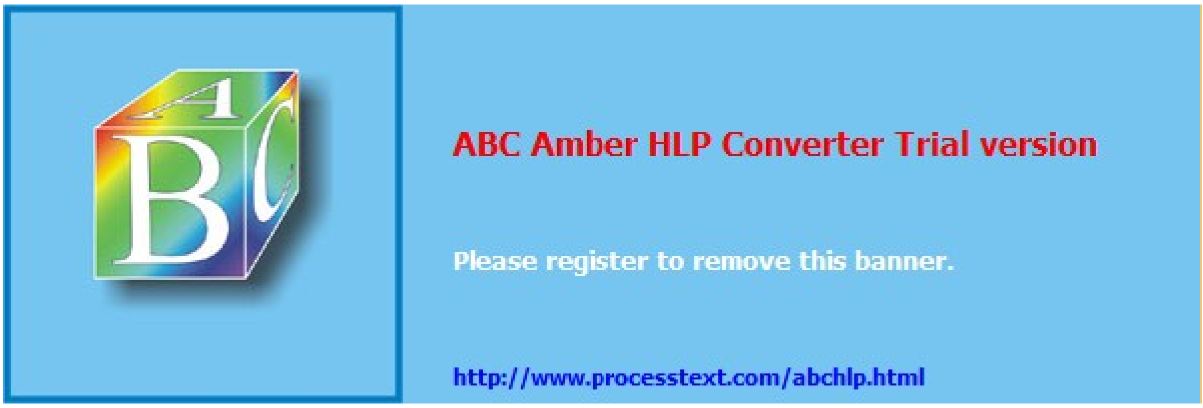
? IBM PC Compatible – 80386DX, 80486DX, Pentium or higher
? 8 MB RAM
? Approx 2.0 MB Free hard disk space
? Microsoft Windows 95/98 or Windows NT4.0 Operating System
? ECP Parallel Port
? Standard serial/PS2 mouse

Connecting The Programmer To Your PC

? Check the contents of your ICE Technology programmer package. Make sure that you have the following:-

1x Device Programmer
1x Copy of programmer software
1x Power supply (UK, Euro, USA or Japanese)
1x Parallel cable (25-pin D Type)
1x User Manual

? Connect one end of the parallel cable to the programmer and the other into one of the parallel ports on the PC. Tighten the holding screws to ensure the cable is properly attached to the LPT port
? Connect the power supply provided into the back of the programmer and plug into any mains socket. NOTE:- The programmer does not have a power on switch. It will power up (the red power LED will light up) automatically when it receives a signal from the PC and then power down when not in use, i.e. when the WinLV software is closed.
? Run WinLV in Windows 95
? Run the Selftest utility in WinLV to confirm that your programmer has been connected correctly and the WinLV software is communicating with the programmer.
? If Selftest does not return any errors then you can start using your programmer and WinLV software. If you are experiencing any problems see Trouble Shooting or contact Technical Support at ICE Technology Ltd.



Trouble Shooting

If you are experiencing problems setting up your programmer with the WinLV software check some of these trouble shooting points:-

▶ PROBLEM 1: WinLV installed OK but programmer will not respond

? Check power supply and batteries

? Check parallel port cable

? Check that the LPT port on the PC is set as an ECP parallel port in the PCs BIOS or Windows 95 settings. ICE Technology's programmers will operate using an enhanced mode LPT but this may cause problems.

? Check for interrupt conflicts. ICE Technology's programmers don't use an interrupt although Interrupt 7 is the standard interrupt allocated to LPT 1 so check if any other application or hardware is using this interrupt value.

? Check for any Dongles in other LPT ports on the PC as these can sometime conflict with the programmers communications with the PC.

▶ PROBLEM 2: WinLV installed on two separate PCs but programmer will only operate on one

? Compare the check points given in problem 1 on both PCs

? Try to set up the PC that will not operate with the programmer by duplicating, as near as possible, the configuration of the PC that does work OK with the programmer.

▶ PROBLEM 3: Installed WinLV on a Laptop PC and connected programmer but programmer remains inoperative

? Check power supply and batteries

? Check parallel port cable

? Check LPT port configuration (i.e.: ECP mode preferable over SPP mode)

? Disable any printer drivers in Windows that are using the same LPT port as the programmer. The printer drives are continuously monitoring the parallel port and so this spurious data being sent back and forth can hinder the communications between the PC and the programmer.

▶ PROBLEM 4: On Start Up WinLV reports "INI File Error – Cannot Find Database Information…"

? Check that the device library subdirectories are installed correctly (e.g.:- C:\WINLV\LIBS\PALS, C:\WINLV\LIBS\MEMORY)

? Check that the library files (*.LIB) are present in the following subdirectories:-

…\\Libs\Memory
…\\Libs\Memory48 (48 pin programmers only)
…\\Libs\Pals
…\\Libs\Pal48 (48 pin programmers only)
…\\Emullibs
…\\CTlibs

? Check the WinLV.INI file to make sure that the subdirectory path statements correspond to the actual location of the LIBS and PARS on your PC.

▶ PROBLEM 5: WinLV installed correctly and running but cannot READ a device (i.e.: READ icon grayed-out)

? REMEMBER – The buffer must be in ***Edit Mode*** and NOT ***View Mode*** before WinLV will allow you to read data from a device into the currently open buffer.

? Check that Edit Mode is enabled using the ***Enable Edit Mode*** option in the ***File Menu***.

Show Me Help On The File Menu !

If you are still experiencing trouble running your programmer, please consult our Installation FAQs .

Installation FAQs

Although the programmer and software have been designed to work with all PCs and associated hardware, from 286 to Pentium III, there are occasions when the programmer will not work straight away. Should you encounter any problems, please just follow the steps below and hopefully these will resolve any issue you encounter.

? ? Ensure that you are running the very latest software. If you have purchased the programmer through one of our distributors, it is quite possible that the programmer has been held in stock for a period of time. The software version number is printed on the disks. To check if this has been superseded, please check our web site on the Internet, at www.icetech.com. This will show the latest software version number and release date, plus a full list of associated updates, changes and amendments.

? ? Currently there is no support for UNIX or LINUX Operating systems. However, LINUX drivers are under development, and will be available at a later date. Other OS platforms are also being considered. Details of these will be available on our web site.

These Frequently Asked Questions cover almost all problems encountered during installation:

Q1. Programmer power is connected, but the power light is not lit.

A1. *The power to the programmer is controlled by the software. This means that power is not wasted if the programmer is being run off batteries. The programmer will also power down automatically during periods of inactivity. Try running the Selftest, or running WinLV, EPROM or PAL. The light should come on as the programmer is initialised.*

Q2. When installing the software in Windows 95/98, the message "*system file already in use* " is reported, and the installation quits.

A2. *This is caused by one of two system files (MFC42.DLL or MSVCRT.DLL) appearing to the set-up file that they are in use, or even newer than that being installed. We provide the latest versions of .these files as released on the Microsoft Developer Network collection. To correct this, restart your PC in MS-DOS mode. Go into the windows system subdirectory (CD C:\WINDOWS\SYSTEM). Rename the two files as MFC42.OLD and MSVCRT.OLD. Restart windows, then install the software. There will then be no problems with installation.*

Q3. When I am in the WinLV software, the READ light is greyed out in the operations menu, and I cannot edit the buffer contents.

A3. *This feature is to protect the contents of the buffer from being modified by accident. To enable READ operations, select FILE/ENABLE EDIT MODE.*

Q4. When running the software (Selftest, WinLV, EPROM or PAL), random communications problems are reported. Printers, and other devices connected to the port have no problems.

A4. *First, check that the programmer is connected properly to the parallel port. Remove any dongles to see if this may be the cause of the problem. Also, check our web page to see if you have the latest software version. The software release notes will indicate if there have been comms problems with any specific device. The programmer works best when connected to an ECP port. On newer PC's, this is selected in the BIOS. Note that on Laptops with an Ir port, selecting IrDA 1.1 could cause ECP mode to become unavailable. Select ECP,* **NOT** *ECP/EPP. We have also seen that corrupt .INI files may cause problems. Delete these, and try running the software again. Also, try the programmer on another PC, to see if the programmer, or the PC are the problem, or possibly try separate I/O card if the problem persists.*

Q5. When running WinLV, the software hangs at the splash screen, or when running PAL or EPROM in DOS, the PC crashes. Printers, and other devices connected to the port have no problem.

A5. *Check that all the items in* ***A4*** *have been adhered to. Also, we have seen that adding C:\ICETECH to the PATH statement in the AUTOEXEC.BAT file can resolve this problem, and also ensure that C:\WINDOWS; C:\WINDOWS\SYSTEM and C:\WINDOWS\COMMAND are also in the PATH statement.*

NOTE: In addition to A4 and A5: For certain branded PC's and Notebooks, separate software utilities are available which may solve the port problems. These are required as certain branded machines do not use standard ChipSets for their parallel ports, and also include power saving and low voltage features on the I/O ports. Check their web pages for the utilities:

Compaq:	http://www.compaq.com/support/files/index.html
Dell:	http://www.dell.com/support/index.htm
Gateway:	http://www.gateway.com/support/product/drivers/index.html

If you are still having problems, just call our Technical Support team, who will be happy to help:

tel: +44 (0)1226 767404 fax: +44(0)1226 370434 email: *support@icetech.com*

Downloading WinLV From The WWW

ICE Technology Website

The ICE Technology Home Page on the World Wide Web (WWW) includes up to date information on all of ICE Technology’s products and services. For example –

? Technical Specifications on all ICE Technology products.
? FREE Software and Upgrades.
? Current prices.
? Device support lists.
? Device package adapters.
? Other information including ICE Technology News and International distributors.

ICE Technology Website address – **Home Page**

ICE Technology Software Upgrades – **Software Upgrades**

Connecting The Programmer To A Network

It is possible to run the WinLV software over a network using an ICE Technology programmer that is connected to a PC on the same network. However, you must make sure that the appropriate WinLV.INI files are modified so that the libraries - LIBS and the parameters - PARS subdirectories are pointed to correctly.

Using Your Programmer Over A Network

? Follow instructions to connect the programmer to your local PC

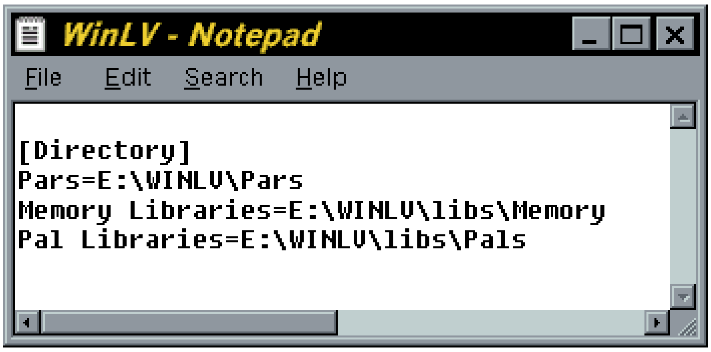
? Install The WinLV software onto one of the network hard drives

? Edit the WinLV.INI file so that the directory paths for Pars, Memory Libraries and Pal Libraries point to the correct subdirectories on the network hard drive.

? Run the Selftest utility in WinLV to confirm that your programmer has been connected correctly and the WinLV software is communicating with the programmer.

WinLV.ini Edit Example

In this example, The E drive on the network holds the Pars and Library information and so the subdirectory paths have been edited as below.



```
[Directory]
Pars=E:\WINLV\Pars
Memory Libraries=E:\WINLV\libs\Memory
Pal Libraries=E:\WINLV\libs\Pals
```

WinLV Screen Arrangement

The WinLV software provides you with an easy-to-use, menu-driven, front-end for ICE Technology's range of Single Socket Multi-Device Programmers and our Speedmaster GLV-32 Gang Programmer. WinLV offers you a comprehensive range of file handling & editing utilities, data manipulation, ROM/RAM Emulator Options, Chiptest Options and device programming functions all incorporated in one "familiar" looking Windows 95 application.

The basic on-screen components of the WinLV software are as follows:-

Programming Buffer Window

The majority of the screen is taken up with displaying the programming buffer . The buffer can be viewed differently by selecting any of the Buffer View options. The buffer can also be edited so that data can be changed before or after it has been programmed into a device.

WinLV Toolbars & Display Panels

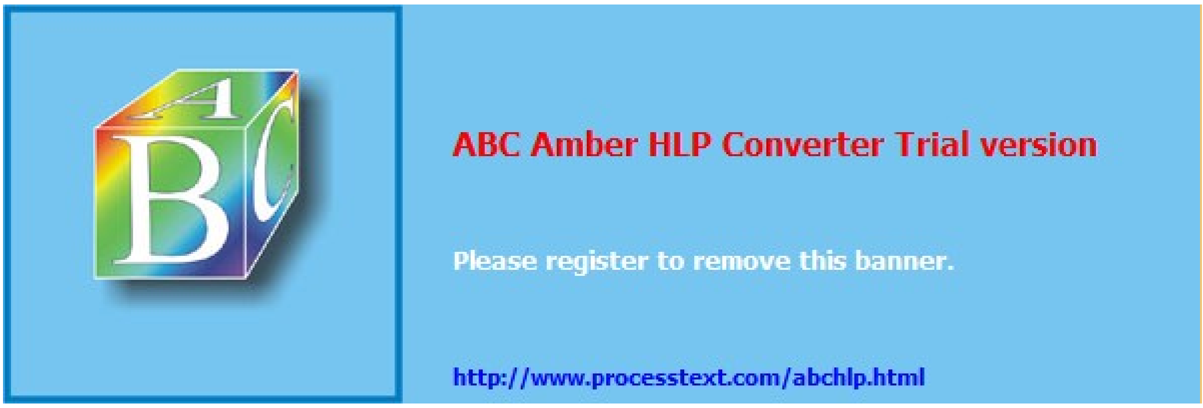
The toolbars & display panels allow you to view current WinLV information and settings as well as select specific functions as an alternative to using the WinLV menus. The Toolbars and Panels are:-

?	General Toolbar	**Show Me Help On General Tool bar !**
?	Status Toolbar	**Show Me Help On Status Tool bar !**
?	Edit Toolbar	**Show Me Help On Edit Tool bar !**
?	Display Mode Panel	**Show Me Help On Display Mode Panel !**
?	Information Panel	**Show Me Help On Information Panel !**

WinLV Menus

The menu selection bar along the top of the screen allows you to view the WinLV menus and then select specific menu items. Use the mouse or keyboard to select the menus and menu commands. The main menus are:-

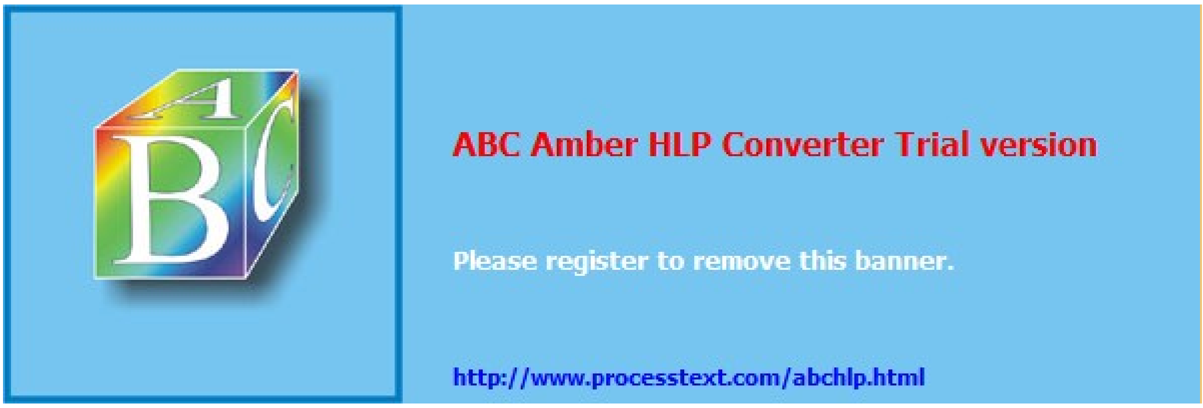
?	File Menu	**Show Me Help On File Menu Options !**
?	Edit Menu	**Show Me Help On Edit Menu Options !**
?	View Menu	**Show Me Help On View Menu Options !**
?	Programmer Menu	**Show Me Help On Programmer Menu Options !**
?	Help Menu	**Show Help On Me Help Menu Options !**



WinLV Menu Options

All commands and programmer actions in WinLV are invoked using the menus. Selecting menus and menu options can be done using a mouse or keyboard. The main function menus are listed below

- **Show Me File Menu Options!**
- **Show Me Edit Menu Options!**
- **Show Me View Menu Options!**
- **Show Me Programmer Menu Options!**
- **Show Me Help Menu Options!**



Tutorial 1: Programming an EPROM from file

This first tutorial is designed to get you operating your programmer and working with WinLV as quickly as possible. It explains, step-by-step, the procedure for programming an EPROM or microcontroller using an existing file. Follow these steps very carefully and soon you will become familiar with the basic principles of the WinLV software and then be able to implement some of the more advanced programming features.

Step 1:- Run WinLV

Select the WinLV.EXE file using the Windows 95 Explorer or, alternatively, create a shortcut on the Windows 95 Desktop.

Step 2:- Select Device

Choose the device that you are using by selecting it from the ***Select Device*** window. To open the ***Select Device*** window – click the left hand mouse button on ***Programmer*** from the top menu bar then click on ***Select Device***. Use the mouse to browse through the Parts Database and select a device from the Programmable Parts section then click on Accept to confirm your choice.

I want to select a device now !

Step 3: Load File into Buffer

Select the file you would like to programme into the selected EPROM or Micro following these steps:-

? Select the ***File*** menu and then ***Open…***

? Select which file you want to load into the buffer . OR type the file name into the ***File Name*** box

? If any files that you expecting to see aren't there try using the ***Files of Type*** drop down menu to select which file type is displayed in the main window

? Use the ***Open as*** drop down menu to select which format the file will be opened as and loaded into the buffer.

? Set the Defaults to pre fill the buffer with 0xFF so that all empty address locations are filled with FF.

? Remember - If you want to specify more advanced file open settings this can be set in the lower half of the ***Open*** dialogue window.

? Remember - If the buffer size is not large enough for the file selected then it will truncate the file before loading. To increase the buffer size select ***New Buffer*** from the ***File*** menu and use the slider bar to increase the buffer size and then re-load the file.

Step 4: Place Device in Programmer

Place the device you've selected to programme into the ZIF socket on the programmer. Make sure that the device is aligned with the bottom of the ZIF socket unless instructed to do otherwise, e.g.:- on The MMLV and the MM1000+ the Microchip PIC12C508/509 microcontroller must be aligned with the top of the ZIF socket.

Step 5: programme the Device

Follow these final steps to finish programming the device:-

? Select ***Operations*** from the ***Programmer*** menu to view the Operations For Device dialogue window.

? Click on the ***Programme*** button in the EPROM ***Operations*** dialogue window -

? The device in the ZIF socket will now be erased, programmed and verified.

I want to programme the device now !

Tutorial 2: Programming an EPROM from Master Device

This second quickstart tutorial explains how to read data from a master device into the buffer and then programme another device with same data. This particular example assumes that the device being read and the other device being programmed are the same. It is possible, of course, to read data from one EPROM/Flash device and programme it into another similar EPROM/Flash device that isn't exactly the same. To do this you must re-select the device type from the Parts Database between reading and programming.

Follow these steps very carefully and they will give you some idea of the basic Read & programme operations that WinLV is capable of. When you feel comfortable with reading and programming devices using basic WinLV operations experiment with some of the more advanced options.

▶ Step 1:- Run WinLV

Select the WinLV.EXE file using the Windows 95 Explorer or, alternatively, create a shortcut on the Windows 95 Desktop.

▶ Step 2:- Select Device

Choose the device that you are using by selecting it from the ***Select Device*** window. To open the ***Select Device*** window – click the left hand mouse button on ***Programmer*** from the top menu bar then click on ***Select Device***. Use the mouse to browse through the Parts Database and select a device from the Programmable Parts section then click on Accept to confirm your choice.

I want to select a device now !

▶ Step 3: Place Master Device in Programmer

Place the device you've selected to programme into the ZIF socket on the programmer. Make sure that the device is aligned with the bottom of the ZIF socket unless instructed to do otherwise.

▶ Step 4:- Read Master Device

Follow these steps to read data from the master device:-

? Select ***Operations*** from the ***Programmer*** menu to view the Operations For Device dialogue window.

? Click on the ***Read*** button in the EPROM ***Operations*** dialogue window -

? The device in the ZIF socket will now be read and all the data will be loaded into the buffer and displayed on the screen.

At this point you can save the data in the buffer by selecting ***Save*** or ***Save Project*** from the File Menu and also change the device selection in the Parts Database if you wish to programme a different device with the file you have just read.

▶ Step 5: Programme the Blank Device

Follow these final steps to programme the blank device with the data from the master device:-

? Select ***Operations*** from the ***Programmer*** menu to view the Operations For Device dialogue window.

? Click on the ***Programme*** button in the EPROM ***Operations*** dialogue window -

? The device in the ZIF socket will now be programmed and verified.

I want to programme the device now !

Tutorial 3: Programming a PAL from file

This third tutorial is designed to get you operating your programmer and working with WinLV as quickly as possible. It explains, step-by-step, the procedure for programming a logic device using an existing file. Follow these steps very carefully and soon you will become familiar with the basic principles of the WinLV software and then be able to implement some of the more advanced programming features.

Step 1:- Run WinLV

Select the WinLV.EXE file using the Windows 95 Explorer or, alternatively, create a shortcut on the Windows 95 Desktop.

Step 2:- Select Device

Choose the device that you are using by selecting it from the ***Select Device*** window. To open the ***Select Device*** window – click the left hand mouse button on ***Programmer*** from the top menu bar then click on ***Select Device***. Use the mouse to browse through the Parts Database and select a device from the Programmable Parts section then click on Accept to confirm your choice.

I want to select a device now !

Step 3: Load File into Buffer (Fuse Map)

Select the file you would like to programme into the selected PAL , PALCE, EPLD etc. following these steps:-

? Select the ***File*** menu and then ***Open…***

? Select which file you want to load into the buffer . OR type the file name into the ***File Name*** box. The buffer will automatically change into Jedec Mode and display the buffer as a Fuse Map .

? If any files that you expecting to see aren’t there try using the ***Files of Type*** drop down menu to select which file type is displayed in the main window

? Use the ***Open as*** drop down menu to select which format the file will be opened as and loaded into the buffer.

? Remember - If the buffer size is not large enough for the file selected then it will truncate the file before loading. To increase the buffer size select ***New Buffer*** from the ***File*** menu and use the slider bar to increase the number of fuses in the fuse map and then re-load the file.

Step 4: Place Device in Programmer

Place the device you’ve selected to programme into the ZIF socket on the programmer. Make sure that the device is aligned with the bottom of the ZIF socket unless instructed to do otherwise.

Step 5: Programme the Device

Follow these final steps to finish programming the device:-

? Select ***Operations*** from the ***Programmer*** menu to view the Operations For Device dialogue window.

? Click on the ***Programme*** button in the PAL ***Operations*** dialogue window -

? The device in the ZIF socket will now be erased, programmed and verified.

I want to programme the device now !

Tutorial 4: Emulating an 8-bit EPROM

This fourth tutorial helps to explain how the WinLV ROM/RAM emulator options work. This tutorial introduces you to the basic steps required to emulate an EPROM in Circuit. Once you are familiar with these you may want to investigate the more advanced emulation features such as the comms channel and the “modify on the fly” comms option. Follow these steps:-

Step 1:- Run WinLV

Select the WinLV.EXE file using the Windows 95 Explorer or, alternatively, create a shortcut on the Windows 95 Desktop.

Step 2:- Select Device To Emulate

The WinLV software will automatically detect if there is an ICE Technology ROM/RAM Emulator connected to your programmer and if so it will display the currently available parts to emulate in the Parts database . Remember that only the emulator parts that are applicable to your emulator will be displayed in the Parts database. Therefore if your are attempting to emulate a 16-bit EPROM with an 8-bit emulator installed then only 8-bit EPROMs will be displayed. Choose the 8-bit EPROM emulator part by selecting it from the ***Select Device*** window. To open the ***Select Device*** window – click the left hand mouse button on ***Programmer*** from the top menu bar then click on ***Select Device***. Use the mouse to browse through the Parts Database and select a device from the Emulator Parts section then click on Accept to confirm your choice.

Step 3:- Load File into Buffer

Select the file you would like to download to the emulator following these steps:-

? Select the ***File*** menu and then ***Open…***

? Select which file you want to load into the buffer . OR type the file name into the ***File Name*** box

? If any files that you expecting to see aren’t there try using the ***Files of Type*** drop down menu to select which file type is displayed in the main window

? Use the ***Open as*** drop down menu to select which format the file will be opened as and loaded into the buffer.

? Set the Defaults to pre fill the buffer with 0xFF so that all empty address locations are filled with FF.

? Remember - If the buffer size is not large enough for the file selected then it will truncate the file before loading. To increase the buffer size select ***New Buffer*** from the ***File*** menu and use the slider bar to increase the buffer size and then re-load the file.

Step 4:- Connect Emulator to Target System

Because this tutorial is illustrating how to emulate an 8-bit EPROM then the emulator IDC cable must connect to the EMULATOR I socket on the back of the programmer with the 32-pin EPROM header end connected onto the target system .

Step 5:- Download To The Emulator

Once you have loaded the data file into the buffer (**Step 3**) and connected the emulator to the target system (**Step 4**) follow these steps to download the data to the emulator:-

? Select ***Operations*** from the ***Programmer*** menu to view the Operations For Device dialogue window.

? Click on the ***Download*** button in the emulator ***Operations*** dialogue window -

Step 6:- Start The EPROM Emulation

Now all the data from the buffer has been downloaded into the emulator hardware the next step is to activate the emulator to start the ROM emulation in the target system.

Do this by clicking on the ***Go*** button in the emulator ***Operations*** dialogue window - GO

Step 7:- Stop The EPROM Emulation

Once the emulator is active you will notice that the Go button changes to a Stop button and other options are made available, e.g.- Generate a reset, “Modify on the fly” comms etc. Once the EPROM has been emulated in the target system and you need to stop the emulator click on the Stop button - STOP .

Tutorial 5: Emulating a 16-bit EPROM

This fifth tutorial leads on from the previous tutorial and explains how the WinLV ROM/RAM emulator options work when emulating 16-bit EPROMs. This tutorial introduces you to the basic steps required to emulate a 16-bit EPROM in Circuit. Once you are familiar with these you may want to investigate the more advanced emulation features such as the comms channel and the "modify on the fly" comms option. Follow these steps:-

▶ Step 1:- Run WinLV

Select the WinLV.EXE file using the Windows 95 Explorer or, alternatively, create a shortcut on the Windows 95 Desktop.

▶ Step 2:- Select Device To Emulate

The WinLV software will automatically detect if there is an ICE Technology ROM/RAM Emulator connected to your programmer and if so it will display the currently available parts to emulate in the Parts database . Remember that only the emulator parts that are applicable to your emulator will be displayed in the Parts database. Therefore if your are attempting to emulate a 16-bit EPROM with an 8-bit emulator installed then only 8-bit EPROMs will be displayed. Choose the 16-bit EPROM emulator part by selecting it from the ***Select Device*** window. To open the ***Select Device*** window – click the left hand mouse button on ***Programmer*** from the top menu bar then click on ***Select Device***. Use the mouse to browse through the Parts Database and select a device from the Emulator Parts section then click on Accept to confirm your choice.

▶ Step 3:- Load File into Buffer

Select the file you would like to download to the emulator following these steps:-

? Select the ***File*** menu and then ***Open…***

? Select which file you want to load into the buffer . OR type the file name into the ***File Name*** box

? If any files that you expecting to see aren't there try using the ***Files of Type*** drop down menu to select which file type is displayed in the main window

? Use the ***Open as*** drop down menu to select which format the file will be opened as and loaded into the buffer.

? Set the Defaults to pre fill the buffer with 0xFF so that all empty address locations are filled with FF.

? Remember - If the buffer size is not large enough for the file selected then it will truncate the file before loading. To increase the buffer size select ***New Buffer*** from the ***File*** menu and use the slider bar to increase the buffer size and then re-load the file.

▶ Step 4:- Connect Emulator to Target System

Because this tutorial is illustrating how to emulate an 16-bit EPROM then both of the emulator IDC cable must connect to the EMULATOR I and EMULATOR II sockets on the back of the programmer. For 16-bit EPROMS you will need to connect the IDC cable from EMULATOR I to the top of the 16-bit emulator EPROM adapter and the cable from EMULATOR II into the bottom of the adapter. Plug the 16-bit EPROM adapter onto the target system .

▶ Step 5:- Download To The Emulator

Once you have loaded the data file into the buffer (**Step 3**) and connected the emulator to the target system (**Step 4**) follow these steps to download the data to the emulator:-

? Select ***Operations*** from the ***Programmer*** menu to view the Operations For Device dialogue window.

? Click on the ***Download*** button in the emulator ***Operations*** dialogue window -

▶ Step 6:- Start The EPROM Emulation

Now all the data from the buffer has been downloaded into the emulator hardware the next step is to activate the emulator to start the ROM emulation in the target system.

Do this by clicking on the ***Go*** button in the emulator ***Operations*** dialogue window - GO

▶ Step 7:- Stop The EPROM Emulation

Once the emulator is active you will notice that the Go button changes to a Stop button and other options are made available, e.g.- Generate a reset, "Modify on the fly" comms etc. Once the EPROM has been emulated in the target system and you need to stop the emulator click on the Stop button - STOP .

Tutorial 6: Emulating a 8-bit RAM

This sixth tutorial provides step by step instructions to emulating a RAM in circuit. The procedure is very similar to the other emulator tutorials for 8-bit and 16-bit EPROMs.

Step 1:- Run WinLV

Select the WinLV.EXE file using the Windows 95 Explorer or, alternatively, create a shortcut on the Windows 95 Desktop.

Step 2:- Select Device To Emulate

The WinLV software will automatically detect if there is an ICE Technology ROM/RAM Emulator connected to your programmer and if so it will display the currently available parts to emulate in the Parts database . Remember that only the emulator parts that are applicable to your emulator will be displayed in the Parts database. Therefore if your are attempting to emulate a 1024Kx8 RAM with an emulator that can only emulate up to 256Kx8 RAMs then only RAMs up to 256Kx8 will be displayed. Choose the emulator part by selecting it from the ***Select Device*** window. To open the ***Select Device*** window – click the left hand mouse button on ***Programmer*** from the top menu bar then click on ***Select Device***. Use the mouse to browse through the Parts Database and select a RAM from the Emulator Parts section then click on Accept to confirm your choice.

Step 3:- Load File into Buffer

Select the file you would like to download to the emulator following these steps:-

? Select the ***File*** menu and then ***Open…***

? Select which file you want to load into the buffer . OR type the file name into the ***File Name*** box

? If any files that you expecting to see aren't there try using the ***Files of Type*** drop down menu to select which file type is displayed in the main window

? Use the ***Open as*** drop down menu to select which format the file will be opened as and loaded into the buffer.

? Set the Defaults to pre fill the buffer with 0xFF so that all empty address locations are filled with FF.

? Remember - If the buffer size is not large enough for the file selected then it will truncate the file before loading. To increase the buffer size select ***New Buffer*** from the ***File*** menu and use the slider bar to increase the buffer size and then re-load the file.

Step 4:- Connect Emulator to Target System

Because this tutorial is illustrating how to emulate an 8-bit RAM then the emulator IDC cable must connect to the EMULATOR I socket on the back of the programmer with the 32-pin EPROM header end connected onto the target system .If the RAM you have selected to emulate has less than 32 pins then align the EPROM header with the bottom of the RAM socket on the target system, all unused pins will be ignored.

Step 5:- Download To The Emulator

Once you have loaded the data file into the buffer (**Step 3**) and connected the emulator to the target system (**Step 4**) follow these steps to download the data to the emulator:-

? Select ***Operations*** from the ***Programmer*** menu to view the Operations For Device dialogue window.

? Click on the ***Download*** button in the emulator ***Operations*** dialogue window -

Step 6:- Start The RAM Emulation

Now all the data from the buffer has been downloaded into the emulator hardware the next step is to activate the emulator to start the RAM emulation in the target system.

Do this by clicking on the ***Go*** button in the emulator ***Operations*** dialogue window - GO

Step 7:- Stop The RAM Emulation

Once the emulator is active you will notice that the Go button changes to a Stop button and other options are made available, e.g.- Generate a reset, “Modify on the fly” comms etc. Once the RAM has been emulated in the target system and you need to stop the emulator click on the Stop button - STOP .

Step 8:- Upload From The Emulating RAM to The Emulator

At any point during the RAM emulation after it has stopped you can upload the data from the RAM into the emulator. All the data uploaded will be displayed in the buffer for inspection.

Click on the ***Upload*** button in the emulator ***Operations*** dialogue window -

---

Selftest

Running The *Selftest* Utility

The WinLV software contains a programmer *Selftest* utility that enables you to check the following:-

? Checks your programmer has been installed correctly

? Checks for communications between PC and programmer

? Check for any internal hardware faults in your programmer

Use the *Selftest* when you connect up a new programmer, move your programmer to a new PC or if you suspect there may be fault with your programmer.

WARNING:- Always ensure that there is no device in the programmer ZIF socket before running *Selftest* . Doing so may lead to damaging the device and incurring false results from the *Selftest* diagnostic

I want to run Selftest now !

What does the *Selftest* Do ?

? Tests communications between the programmer and PC via the parallel cable

? Tests programmer firmware version

? Tests programmer ROM and reports checksum

? Tests programmer RAM

? Tests programmer DACs

? Tests programmer ZIF socket pin drivers

Emulator Test

Running *The Emulator Test* Utility

The WinLV software contains an ***Emulator Test*** utility that enables you to check the following:-

? Communications between programmer and emulator

? Emulator hardware including emulator on-board RAM

Use the *Emulator Test* when you install a new emulator into a new programmer, or move the emulator from one programmer to another or if you suspect there may be fault with your emulator.

I Want To Run The Emulator Test Now !

File Menu

The WinLV File Menu contains the standard Windows commands for opening, saving & printing files.

▶ New Buffer (Ctrl+N)

Allows you to open and specify the size of the programming buffer that WinLV is using. The minimum buffer size is 256K and the maximum is 32M. To alter the buffer size use the mouse or keyboard to move the buffer size selector to increase or decrease the amount of buffer memory.

If you are programming a logic device, e.g. PAL , GAL or EPLD etc. the ***Buffer Size*** window allows you to specify how many Jedec fuses are in the buffer. To alter the number of fuses use the mouse or keyboard to move the Jedec fuses selector to increase or decrease the size of the fuse map .

▶ Open… (Ctrl+O)

Allows you to open previously saved files into the buffer. The ***Open*** dialogue window is set out using the normal, familiar, Windows 95 arrangement.

To open a file:-

? Select the ***File*** menu and then ***Open…***

? Use the mouse or keyboard to select which file you want to load into the buffer. OR type the file name into the ***File Name*** box

? Use the ***Files of Type*** drop down menu to select which file type is displayed in the main window

? Use the ***Open as*** drop down menu to select which format the file will be opened as and loaded into the buffer.

? Set the Defaults to pre fill the buffer with 0xFF if you are opening a new file. You can change the value used to fill the buffer and the length of each value in the ***Set Defaults*** section.

The lower section of the ***Open*** dialogue window also allows you to select other advanced file loading options (Not available when opening *.JED files for logic devices):-

? The ***Load Every*** option allows you to select if every byte, every 2nd byte, every word, every 2nd word etc. is loaded from the file into the buffer.

? ***Between address*** allows the lower and upper address to be set for reading a file into the buffer.

? ***Store into every*** selects if you want to load the file data into every byte, every 2nd byte etc.

? ***Starting at buffer address*** allows you to specify an address in the buffer where the first address from the file will be loaded.

? ***Swap Bytes (within words)*** and ***Swap Words (within long)*** allow the MSB and LSB to be swapped when loading a file into the buffer.

▶ Save (Ctrl+S)

Allows you to save currently open files from the buffer in any file format. Use the mouse or keyboard to select a file name and file type.

▶ Save As…

Allows you to save a currently opened file in the buffer with a different file name. The ***Save As…*** dialogue window looks much like the ***Save*** and ***Open*** windows and operates in an identical way.

▶ Open Project

Allows you to open a previously saved project file with a *.ICE extension and load all the information stored in the project into WinLV.

▶ Save Project

Allows you to save all of the currently selected WinLV options into one project file (*.ICE)

▶ Print… (Ctrl+P)

Allows you to print the currently open buffer to either a local or network printer. Selecting ***Print..*** will open a dialogue window in which you can select:-

- Which printer (local or network) and printer properties.
- Print range, i.e.: specify a range of pages to be printed by giving the start page number and finish page number.
- How many copies

The printed buffer will display addresses and the hexadecimal data stored in the addresses, or if the buffer is set up for a logic device, it will display fuse numbers and the fuse map .

▶ Print Preview

Allows you to preview the currently open buffer before it is printed. The ***Print Preview*** menu option displays the pages of the current buffer as they will appear after they have been printed.

▶ Print Setup…

Allows you to select a default printer for WinLV to use and also define the current printer properties. Other printer options, such as paper size and orientation, can be selected using ***Print Setup…***.

▶ Recent Files

Allows you to re-open a previously saved file. The last, most recent, file names that you have been using with WinLV will be stored here.

▶ Recent Projects

Allows you to re-open a previously saved project . The last, most recent, project file names that you have been using with WinLV will be stored here.

Enable View Mode Only / Enable Edit Mode

This option allows you to toggle between editing and viewing the buffer . The ***View Mode Only*** will prompt you for a password that must be entered when you select to go back into ***Edit Mode***. If you do not wish to enter a password to return to ***Edit Mode*** then click on ***OK*** twice in the password dialogue window and no password will be saved.

In ***View Only Mode*** the user will not be allowed to perform operations or change parameters that affect the contents of the buffer – such as reading, editing or serial numbering. The main window will be hidden in this mode, with only the ***Programmer Operations*** window open.

The setting of this mode can be saved within a Project File , which can be opened by a user by just double-clicking on the project icon. In this case all other programming parameters are also saved so once the project is opened, the programming can commence with no input from the operator other than clicking the “Programme” button or, in hands free mode, replacing the devices.

Exit

Select ***Exit*** to close the WinLV software and power down your programmer.

Edit Menu

The WinLV Edit Menu contains commands for easily manipulating the data displayed in the programming buffer. It Includes utilities for finding specific data in the buffer as well as byte/word swap and buffer fill and others.

Remember - The functions available in the Edit Menu will sometimes change depending on whether you are programming a device, using the Chiptest Utility or the ROM/RAM Emulator.

▶ Copy (Ctrl+C)

This menu option allows highlighted sections of the programming buffer to be copied to a different location (address) in the buffer.

To copy a selection of addresses from one location in the buffer to another:-

? Move the mouse pointer to the start address of selection.

? Hold down the left mouse button whilst dragging the pointer across the buffer to include all address to be copied.

? Release the left mouse button when the mouse pointer is above the last address in the selection to be copied.

? Without touching the mouse buttons or pressing any other keys, select ***Copy*** from the ***Edit*** menu.

? Move the mouse pointer and click once on the start address in the buffer where you want to copy the highlighted selection.

? Select ***Paste*** from the ***Edit*** menu and the highlighted selection will be pasted into the new location in the buffer.

▶ Paste (Ctrl+V)

This allows you to paste a copied, highlighted selection from one location in the buffer to another. (See ***Copy***)

▶ Goto (Ctrl+G)

This allows you to locate and go to a specific address in the buffer. Enter the hexadecimal byte or word address in the ***Goto*** dialogue window and the buffer view will change to display the new location.

▶ Find (Alt+F3)

This allows you to find a specific hexadecimal, octal or ASCII value in the currently open buffer. Select one the hexadecimal, octal or ASCII radio buttons and enter the value to search for in the ***Find String*** box.

I want to find a value in the buffer now !

▶ Find next (F3)

After you have specified a search string using the ***Find*** menu option use ***find next*** to search through the buffer and locate the next occurrence of the search string.

▶ Find previous (Shift+F3)

After you have specified a search string using the ***Find*** menu option use ***find previous*** to search back through the buffer , from currently selected address, and locate the previous occurrence of the search string.

▶ Fill (Ctrl+F)

Allows you to fill the currently open buffer with a specified hexadecimal value. The ***Fill*** dialogue window allows you to enter a value (as a Byte or Word) and the start and end address within which all addresses will be filled with the selected value. The ***Fill*** default start and end addresses are set to the size of the currently open buffer.

I want to fill the buffer now !

▶ Swap Bytes

Allows you to swap the MSB and LSB within any word in the currently open buffer. ***The Swap Bytes within Word*** dialogue window allows you to select the start and end addresses of an area of the buffer within which all word bytes will be swapped.

I want to swap bytes now !

▶ Swap Words

Allows you to swap the Most Significant Word and Least Significant Word within any Long value in the currently open buffer. ***The Swap Words within Long*** dialogue window allows you to select the start and end addresses of an area of the buffer within which all words will be swapped.

I want to swap words now !

▶ Checksum

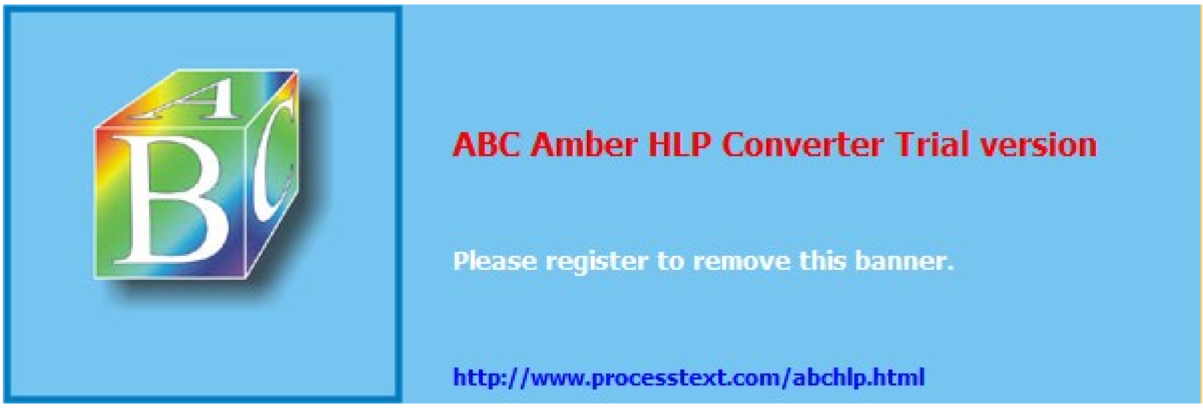
Allows you to calculate a buffer checksum (using the checksum calculator) or a device checksum . The checksum calculator can calculate a checksum for a user defined range of addresses in the buffer, or for the complete buffer. Checksums calculated are:-

? The sum of unsigned bytes with the carry being added to the MSB

? 1's Complement (Inverse of the above checksum)

? CRC – Cyclic Redundancy Check

Show me the Checksum Calculator !



View Menu

The WinLV View Menu contains commands for hiding/showing toolbars and changing the view mode of the buffer.

▶ Toolbar

Allows you to show or hide the General Toolbar on the screen by clicking the left mouse button. The checkmark indicates that the toolbar is active and not hidden. To show a hidden toolbar, select the relevant toolbar from the ***View*** menu and click once with left mouse button.

▶ Operations Toolbar

Allows you to show or hide the Programmer Operations Toolbar on the screen by clicking the left mouse button. The checkmark indicates that the toolbar is active and not hidden. To show a hidden toolbar, select the relevant toolbar from the ***View*** menu and click once with left mouse button.

▶ Edit Toolbar

Allows you to show or hide the Edit Toolbar on the screen by clicking the left mouse button. The checkmark indicates that the toolbar is active and not hidden. To show a hidden toolbar, select the relevant toolbar from the ***View*** menu and click once with left mouse button.

▶ Display mode panel

Allows you to show or hide the Display Mode Panel on the screen by clicking the left mouse button. The checkmark indicates that the display panel is active and not hidden. To show a hidden display panel, select the relevant display panel from the ***View*** menu and click once with left mouse button.

▶ Information panel

Allows you to show or hide the Information Panel on the screen by clicking the left mouse button. The checkmark indicates that the display panel is active and not hidden. To show a hidden display panel, select the relevant display panel from the ***View*** menu and click once with left mouse button.

▶ Split

Allows you to view the currently open buffer using two separate windows. Each window can display a different area in the same buffer.

To split the screen:-

? Select ***Split*** from the ***View*** menu.

? Drag the mouse point to position on the screen where you would like the split and press the left mouse button.

? Edit / view the two areas of the same buffer independently.

? Close the split by moving the mouse pointer over the bar between the two buffer windows, hold down the left mouse button, and dragging the bar up to the top of the buffer window. Alternatively, you can move the mouse pointer over the bar between the two split screens and double-click.

Synchronize split

If you are using the ***Split*** option to view the buffer using two separate windows, the ***Synchronize split*** menu option will allow you to view / edit one of the windows in a different view mode , with any changes to the buffer being displayed in both split windows. The checkmark indicates that the synchronize split option is enabled.

I want to synchronize the split buffer now !

Colours

Allows you to change the default colour settings for WinLV.

I want to change the colours now !

Nibble Mode

Only applicable when programming EPROMs and Flash memory etc. ***Nibble Mode*** displays the current buffer using 16 nibbles per line. The nibbles are arranged so that the LSN is on the left hand side and the MSN is on the right hand side of each pair of nibbles.

Show me Nibble Mode !

Byte Mode

Only applicable when programming EPROMs and Flash memory etc. ***Byte Mode*** displays the current buffer using 2 bytes (16 bits) on one line. Therefore, buffer addresses down the left hand side start at 00h and increase in 16 bit intervals, e.i:- 00h 10h 20h 30h 40h…F0h etc. The ASCII equivalent characters at each address in the buffer are displayed down the right hand side of the buffer.

Show me Byte Mode !

---

12 Bit Mode

Only applicable when programming EPROMs and Flash memory etc. ***12 Bit Mode*** displays the current buffer using 12 bits (3 nibbles) at buffer address, with 8 address on one line. Therefore, buffer addresses down the left hand side start at 00h and increase in 8 bit intervals, i.e.:- 00h 08h 10h 18h 20h…F8h etc.

Show me 12 bit Mode !

Word Mode (Even:Odd)

Only applicable when programming EPROMs and Flash memory etc. ***Word Mode (Even:Odd)*** displays the current buffer using 8 2-byte words on each line. Each word is displayed with the even byte on the left-hand side and the odd byte on the right-hand side. Buffer addresses down the left hand side start at 00h and increase in 8 bit intervals, e.i:- 00h 08h 10h 18h 20h…F8h etc.

Show me Word (Even:Odd) Mode !

Word Mode (Odd:Even)

Only applicable when programming EPROMs and Flash memory etc. ***Word Mode (Odd:Even)*** displays the current buffer using 8 2-byte words on each line. Each word is displayed with the odd byte on the left-hand side and the even byte on the right-hand side. Buffer addresses down the left hand side start at 00h and increase in 8 bit intervals, e.i:- 00h 08h 10h 18h 20h…F8h etc.

Show me Word Mode (Odd:Even) !

---

8 Bit Octal Mode

Only applicable when programming EPROMs and Flash memory etc. ***Octal Mode*** displays the current buffer using buffer addresses and data in octal.

Show me Octal Mode !

Jedec Mode

Only applicable when programming logic devices e.g.: PALs , GALs etc. ***Jedec Mode*** displays the current buffer as a Jedec compatible fuse map

Show me Jedec Mode !

Vector Mode

Only applicable when programming logic devices e.g.: PALs , GALsetc. Selecting ***Vector Mode*** allows you to view and edit the test vectors applicable to the currently selected logic device.

Show me Vector Mode !

Waveform Mode

Only applicable when programming logic devices e.g.: PALs , GALsetc. ***Waveform mode*** displays the currently open fuse map showing the logic waveforms generated at each pin.

To edit the waveforms at each pin:-

? Select ***Waveform Mode*** from the ***View Menu***

? Select a pin number from the left hand side (move the cursor using the mouse)

? Use the following keys to edit the wave forms at each pin:-

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 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.

Show me Waveform Mode !

Programmer Menu

The WinLV Programmer Menu contains commands for controlling the operations of the programmer, selecting devices and running diagnostic selftests.

Remember:- If you are using the Speedmaster GLV-32 gang programmer with WinLV then the options in the ***Programmer Menu*** will change slightly.

Show Me Help On Programmer Menu (Gang Programming) !

Select Device

The ***Select Device*** menu item allows you to select which device you are currently using with your programmer. All physical devices including PALs, EPROMs & flash etc. can be selected from the ***Select Device*** dialogue window, as well as emulated devices and chip test parts.

Operations

Selecting the ***Operations*** menu item will open the ***Programmer Operations*** window so that you can control exactly what the programmer is doing. There are five different types of ***Programmer Operations*** depending on which type of device you have currently selected to use from the Parts Database and which programmer is connected to your PC, they are:-

? Programmer Operations for EPROMs/Flash/Microcontrollers

Show Me Help On These Operations !

? Programmer Operations for PALS/GALs/EPLDs

Show Me Help On These Operations !

? Programmer Operations for ROM/RAM Emulator

Show Me Help On These Operations !

? Programmer Operations for Chiptest

Show Me Help On These Operations !

? Programmer Operations for Gang programming

Show Me Help On These Operations !

Match 74/4000 device

Selecting this menu item will open the Programmer Operations for Chiptest and automatically start to match any device in the ZIF socket to any device included in the Chiptest libraries (CTPars). This option is useful for identifying the functionality of an unknown device.

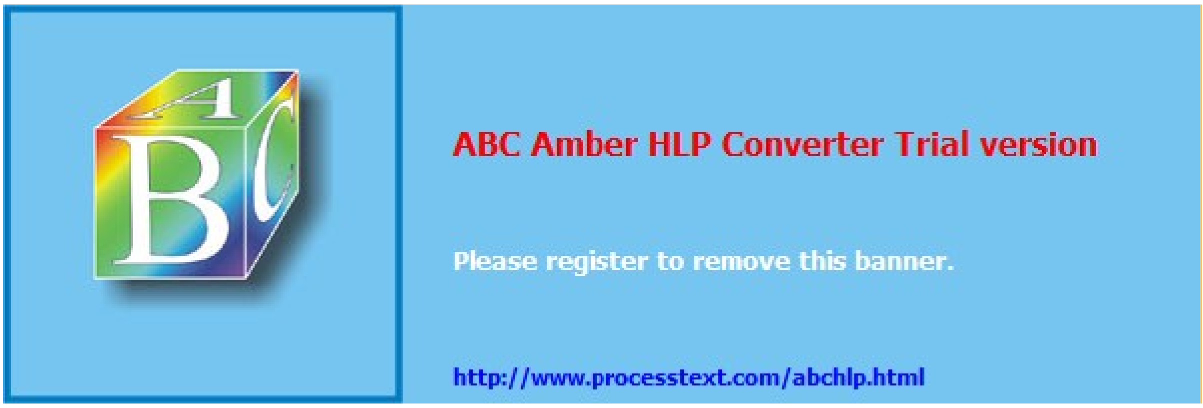
Show Me Help On The *Match 74/4000 Device* option !

Diagnostics

Selecting ***Diagnostics*** allows you to run either:-

? Programmer Selftest **Show Me Help On Selftest !**

? Emulator Test **Show Me Help On Emulator Test !**



Help Menu

The WinLV Help Menu allows you to run the WinLV Help topics and see release information.

▶ Help Topics

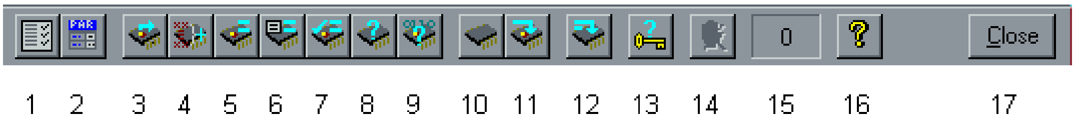
Selecting this runs the WinLV Help topics.

▶ About WinLV…

Selecting this will display the WinLV software version details for your copy of WinLV. Also ICE Technology copyright information.

Programmer Operations for EPROMS/Flash/Microcontrollers

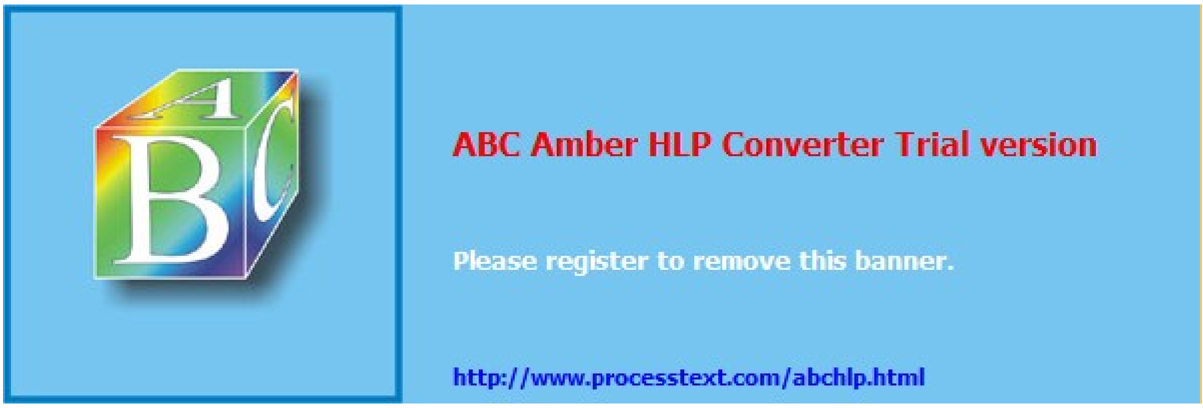
Programmer Operations Toolbar (EPROMs/Flash/Microcontrollers etc.)



Programmer Operations Icons

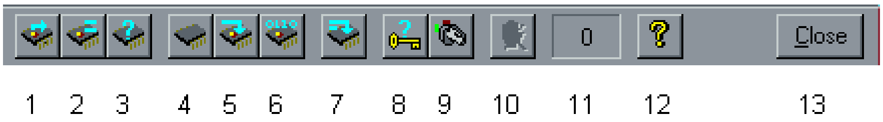
1. Displays ***Programming Options*** window. Allows all programming parameters to be changed.

2. Toggles the ***Force Re-entry Of Parameters*** option.
3. Read device. This operation reads back data already programmed into the device in the ZIF socket.
4. Display ***Device Checksum*** window. Allows device checksum parameters to be changed.
5. Verify device. This operation will verify all of the data that is current held in the buffer with data that has been programmed into the device in the ZIF socket.
6. Verify encryption . This operation verifies the section of the buffer that holds the encryption data with the corresponding encryption array in the device.
7. Verify device signature .
8. Blank check device. This operation checks every address on the device and will report back if it is completely blank (empty) or if it has data at any of its addresses.
9. Bit test device. This operation checks that the data held in the buffer can be successfully programmed into the selected device. This is useful when over-programming a partly programmed device.
10. Erase device. This operation will clear/erase data held in the currently selected device. Empty address locations are usually represented with FF. This operation will only work on erasable devices, e.g. Flash memory
11. Over-Program device. Programs & verifies the device with the data held in the buffer without performing the blank check or bit test priory to the programme cycle.
12. Program device. Programs and verifies the currently selected device.
13. Reports back any device security information.
14. “Hands Free Programming” (*Only Available on the 48 pin programmers*). This operation helps you to programme large amounts of the same device. Once selected, follow the on screen instructions.
15. Program counter display. Keeps a count of how many of the currently selected devices have been programmed.
16. WinLV Help
17. Close. Select this to close the ***Programmer Operations*** window.



Programmer Operations for PALs/GALs/EPLDs

Programmer Operations Toolbar (PALs/GALs/EPLDs etc.)

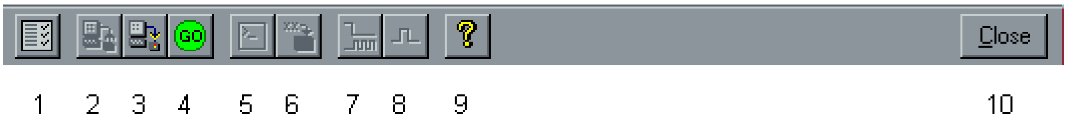


Programmer Operations Icons

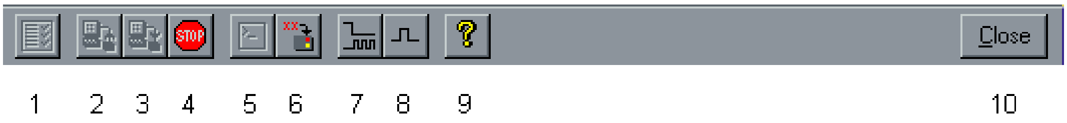
1. Read device. This operation reads back data already programmed into the device in the ZIF socket.
2. Verify device. This operation will verify all of the data that is current held in the fuse map with data that has been programmed into the device in the ZIF socket.
3. Blank check device. This operation checks every fuse on the device and will report back if it is completely blank (usually represented as all fuses set to 1) or if it is not blank.
4. Erase device. This operation will clear/erase data held in the currently selected device.
5. Over-Program device. Programs the device with the data held in the fuse map buffer.
6. Vector Test. This operation allows you to apply the test vectors for the currently selected device.
7. Erase, Program, Verify & Vector test. This operation performs a complete erase, program, verify and vector test on the currently selected device.
8. Reports back any device security information.
9. Auto-secure. This operation toggles the auto-secure feature to ON or OFF. If the auto secure is set to ON then the device to be programmed will be automatically secured.
10. “Hands Free Programming” (*Only Available on the 48 pin programmers*). This operation helps you to programme large amounts of the same device. Once selected, follow the on screen instructions.
11. Program counter display. Keeps a count of how many of the currently selected devices have been programmed.
12. WinLV Help.
13. Close. Select this to close the ***Programmer Operations*** window.

Programmer Operations for the ROM/RAM Emulator

Programmer Operations Toolbar (ROM/RAM Emulator)- *Emulator Not Active*



Programmer Operations Toolbar (ROM/RAM Emulator)- *Emulator Active*

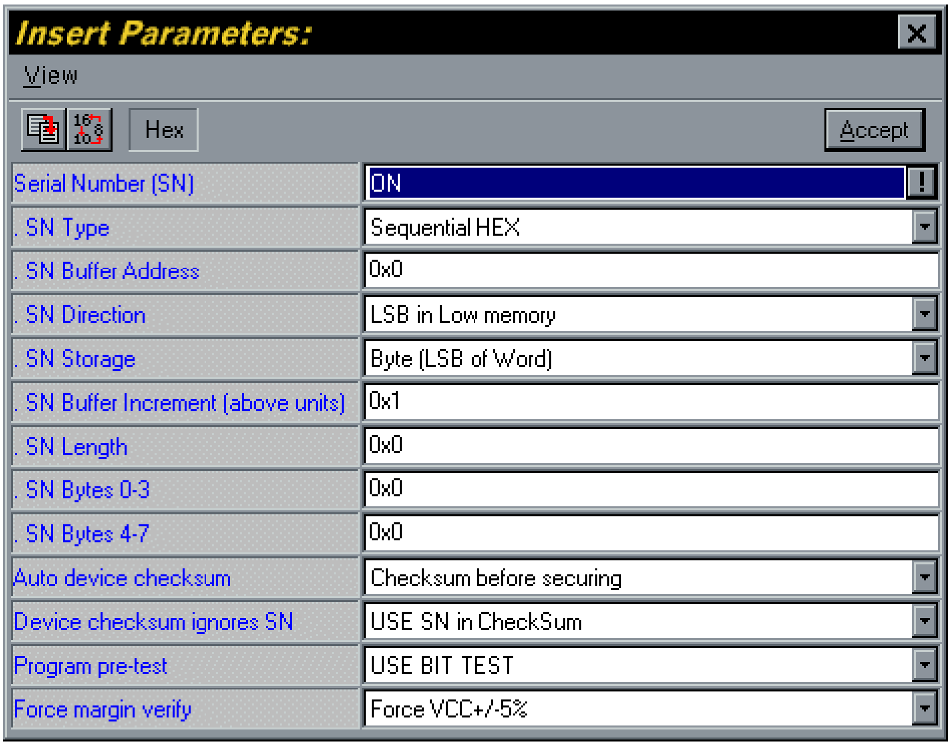


Programmer Operations Icons

1. Displays ***Emulator Parameters*** window. Allows all emulator parameters to be changed.
2. Upload. This operation allows you to upload data from the emulating ROM or RAM in the target system All the data uploaded will be displayed on screen in the programming buffer
3. Download. This operation is the reverse of previous one. It allows you to download the data currently loaded into the buffer into the emulator hardware.
4. Go / Stop. This will either set the emulator status to active or stopped.
5. Open Comms.
6. “Modify on the fly”.
7. Generate a reset. This operation will generate a manual reset to the target system. Pins 1 to 23 on the ZIF Socket can be used as an active low reset signal and pins 25 to 43 can be used as an active high reset signal. To generate a reset make sure that you connect any of these ZIF socket pins to the reset line on the target system.
8. Generate an interrupt.
9. WinLV Help.
10. Close. Select this to close the ***Programmer Operations*** window.

Programming Options / Parameters

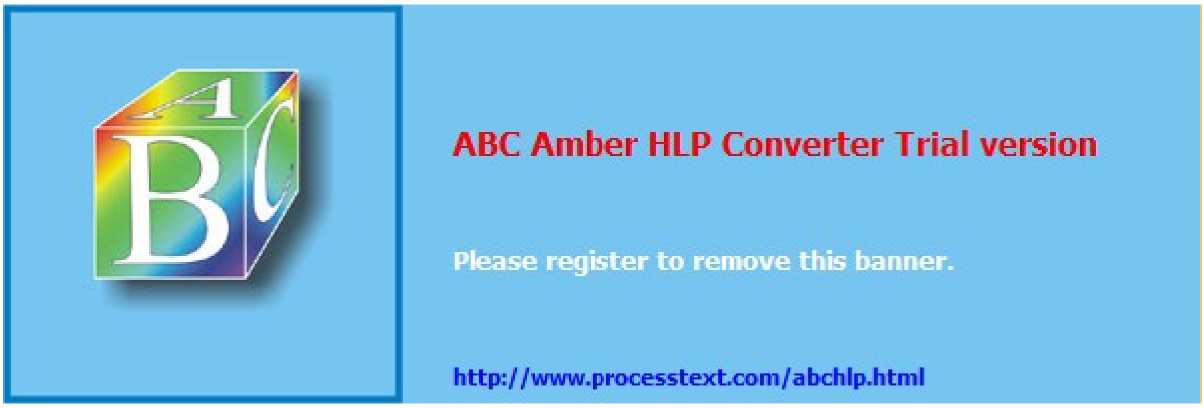
Programming Parameters for EPROMs/Flash/Microcontrollers etc.



Serial Number (SN)	ON
. SN Type	Sequential HEX
. SN Buffer Address	0x0
. SN Direction	LSB in Low memory
. SN Storage	Byte (LSB of Word)
. SN Buffer Increment (above units)	0x1
. SN Length	0x0
. SN Bytes 0-3	0x0
. SN Bytes 4-7	0x0
Auto device checksum	Checksum before securing
Device checksum ignores SN	USE SN in CheckSum
Program pre-test	USE BIT TEST
Force margin verify	Force VCC+/-5%

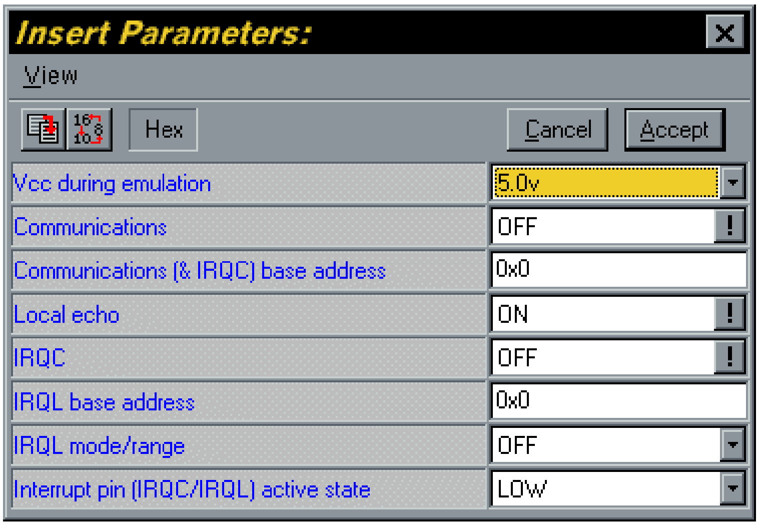
1. Serial Number (SN). Select serial number option ON or OFF.
2. SN Type. Select what format the serial number should take.
3. SN Buffer Address. Select at which address in the buffer should the serial number be stored.
4. SN Direction. Select LSB or MSB in low memory. E.g. a-byte serial number starting at 0000 at buffer address 00 being incremented by 1 every time a device is programmed. Selecting LSB will increment address 00 to FF with address 01 being the overflow. Selecting MSB means that address 03 is incremented to FF with address 02 being the overflow.
5. SN Storage. Select between nibble, byte or word.
6. SN Buffer Increment. Select the value to increment every serial number.
7. SN Length. Select the number of bytes to be used for storing the serial number.
8. SN (bytes 0-3). Select Least significant 4 bytes (8 digits).
9. SN (bytes 4-7). Select Most significant 4 bytes (8 digits).
10. Auto-device Checksum. Select before or after securing the device.
11. Select which pre-program test to carry out.
12. Force Margin Verify. After the programme cycle an automatic verify cycle is performed. This is at the nominal operating voltages as specified by the device manufacturer. The margin verify will verify the currently selected device at +/- 5% of the nominal verify voltage.

Show Me Help On Programmer Operations !



Emulator Options

ROM/RAM Emulator Parameters



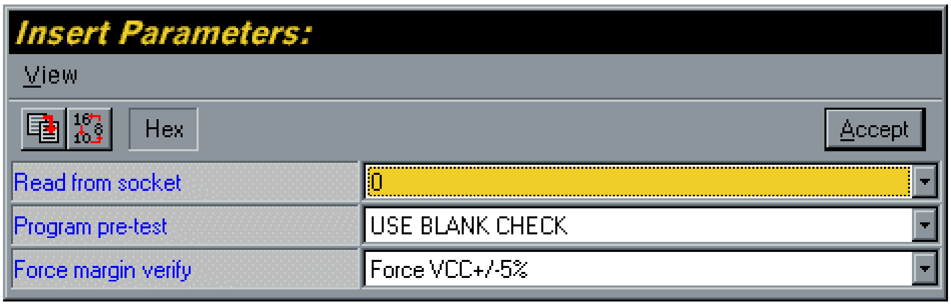
Vcc during emulation	5.0v
Communications	OFF
Communications (& IRQC) base address	0x0
Local echo	ON
IRQC	OFF
IRQL base address	0x0
IRQL mode/range	OFF
Interrupt pin (IRQC/IRQL) active state	LOW

1. Vcc During Emulation. Select either 5V or 3V.
2. Communications ON or OFF
3. IRQC. Select this to enable or disable the interrupt options during emulator comms.
4. Local Echo ON or OFF
5. IRQC Base Address. Select the base address for the segment in the memory that generates an interrupt when accessed during emulator comms.
6. IRQL Base Address. Select the base address for the segment in the memory that generates an interrupt when accessed.
7. IRQL mode /range. Select the range of addresses from the IRQL base address that will generate an interrupt.
8. Interrupt pin active state. Select whether the interrupt signal should be a logic high or logic low.

Show Me Help On Emulator Programmer Operations !

Gang Programming Options / Parameters

Gang Programming Parameters



Read from socket	0
Program pre-test	USE BLANK CHECK
Force margin verify	Force VCC+/-5%

1. Select which ZIF socket data is to be read from
2. Select which pre-program test to carry out.
3. Force Margin Verify. After the programme cycle an automatic verify cycle is performed. This is at the nominal operating voltages as specified by the device manufacturer. The margin verify will verify the currently selected device at +/- 5% of the nominal verify voltage.

Emulating RAMs

What Is RAM Emulation ?

RAM emulation is the process of simulating the functionality of a Random-Access Memory in circuit. To emulate a RAM in-circuit is to completely do away with the need for a real, physical memory IC (Integrated Circuit). You can develop your code and test its performance in system by using the RAM emulator. Because the emulator can download code and also upload code from the target system this provides you with the extremely useful tool for viewing the state of the code in an emulating RAM at any time. Because RAMs can be written to you can observe what data has been changed in the RAM using the emulator. E.g.- viewing stack conditions and registers etc.

Show Me Tutorial On Emulating RAMs !

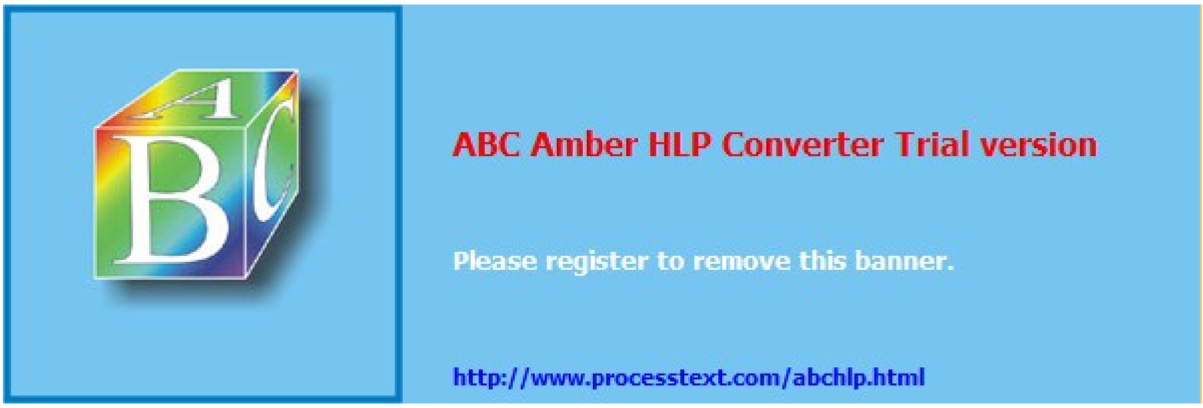
Show Me Help On Emulator Operations !

Emulating ROMs

What is ROM Emulation ?

ROM emulation is the process of simulating the functionality of an EPROM in circuit. To emulate a ROM in-circuit is to completely do away with the need for a real, physical memory IC (Integrated Circuit). You can develop your code and test its performance in system by using the emulator. More importantly, you can speed up development time by emulating an EPROM rather than using its UV-erasable equivalent. Any changes or revision can be done almost instantly using the emulator and the effects of these changes can be observed in-circuit quickly and easily.

Show Me Tutorial On Emulating 8-bit EPROMs !
Show Me Tutorial On Emulating 16-bit EPROMs !
Show Me Help On Emulator Operations !



Emulator Comms - The Comms Channel

The Comms Channel

The first 16 address bytes of the comms channel from COMMSBASE to COMMSBASE + 0Fh are used for sending and receiving data from the PC to the target system. For example, if the emulating ROM routine required that various numbers be sent as inputs from the keyboard then each number would be sent using its ASCII equivalent. Therefore, if the number “1” was sent from the PC to the target, it would sent as ASCII “31”, the number 2 as ASCII “32” and so on. . The comms protocol works in such away that when address COMMSBASE + 0 is read a “0” is sent as a nibble, when COMMSBASE + 1 is read a “1” is sent as a nibble and so on up to COMMSBASE + 0Fh. Therefore, because the comms channel can send any nibble in the range of 00h to 0Fh, all byte permutations are possible.

Comms Channel Reset

The comms protocol also includes a “time-out” routine that will check for the possibility of transmission errors during the emulation comms mode. Basically, the PC will wait until the first nibble (LSN) is sent from the target system. Once this has been received the PC will expect the next nibble (MSN) to be sent within 1ms of receiving the first. If the second nibble does not arrive during this 1ms gap then the PC assumes that there has been an error and will continue to wait for the same first nibble to be re-sent. This 1ms gap means that using a 2ms time delay at the target end before any emulator communications commences will act as a comms reset and initialise (synchronise) the data being transmitted.

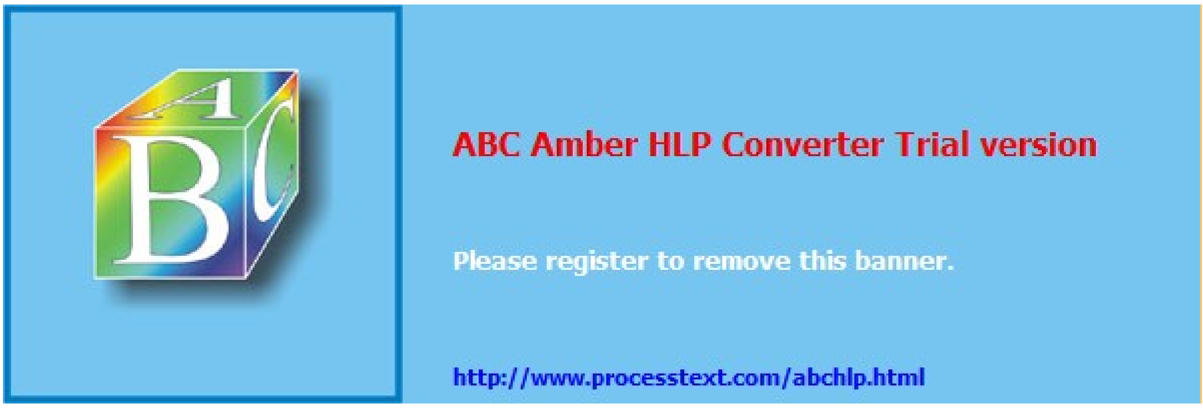
Emulator Comms - The COMMSBASE

The Emulator COMMSBASE

The COMMSBASE is one of the main components of the emulator comms. It is actually a constant, user-defined value that indicates at which start address in memory the comms channel is set up at. Once the COMMBASE has been defined the next 32 addresses starting at the address specified in COMMSBASE will be reserved for use with the comms channel only. Therefore, all data passed between the PC and the target system will be done so using only this area of the code memory.

Setting The COMMSBASE

It is very important to remember that the COMMSBASE should be defined at an address that is within the memory address range of the EPROM to be emulated. Thus, taking into consideration the addresses from COMMSBASE to COMMSBASE + 20h. For example, if the emulator is configured as a 2K EPROM (e.g.: 27C16) which has an address range from 00h to 7FFh, then the COMMSBASE address must not exceed 7DFh (7FFh – 20h).



Emulator Comms - Status Register

What Does The Status Register Do ?

The status register is at address COMMSBASE + 10h. This register holds information relating to whether or not the data is being transmitted or received from the target system. The status register consists of the first 3 bits of address COMMSBASE + 10h.

RxFULL, TxFULL & WREN:-

? RxFULL:- If this bit is set (1) it indicates that the PC has sent a byte to be read by the emulating ROM in the target system. If this bit is not set (0) then there is no data currently available to be read by the target.

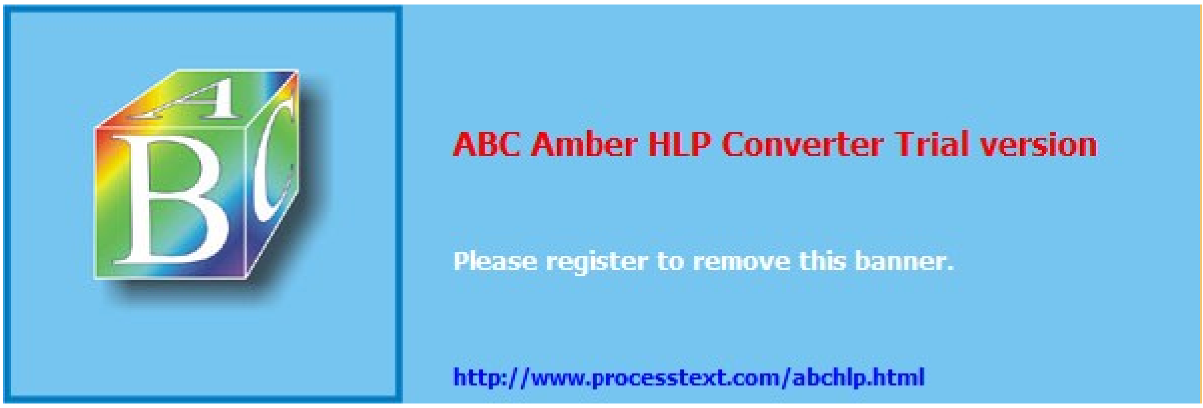
? TxFULL:- This bit must be 0 before a byte can be sent from the emulating ROM in the target system to the PC. If this bit is set then no attempt should be made to read from any address between COMMSBASE to COMMSBASE + 0Fh. When the PC has read each byte that it receives the emulator software will automatically clear TxFULL.

? WREN:- This bit does not have too much affect on the emulator comms. It is actually used to Indicate when an address is being modified by implementing the “modify on the fly” communications (discussed later in this chapter). If WREN is 1 then the PC is modifying an address, if it is 0 then no address is being modified.

Emulator Comms - Receive Register

What Does The Receive Register Do ?

The receive register is at address COMMSBASE + 11h This register holds the current byte of data that is ready to be sent to the PC from the target system. Before any data is held in the receive register the emulating code must do a check to confirm that bit 2 of the status register (RxFULL) is clear otherwise problems can occur.



Emulator Comms - "Modify On The Fly"

What Is “Modify On The Fly” ?

This is process by which data in an emulating ROM can be changed while the emulator is active. It is a complex activity due to the fact that ROMs are read only and so the procedure that changes the data in the emulating ROM does not actually specifically write data. The “Modify on the fly” comms is very useful for changing data in an emulating system without stopping the emulator. The data (one byte at a time) can be downloaded to a user-defined address to the ROM in the target system while the emulator is active.

How To “Modify On The Fly”

Follow theses step to “Modify on the fly”:-

? Start emulating an EPROM using the emulator (Remember – “Modify on the fly” is only applicable on ROMs and not RAMs).

? Click on the ***Modify on the fly*** button on the emulator ***Operations*** dialogue window -
? Specify the address to be changed and the new value.

The “Modify on the fly” operation works by holding the new value in the emulator RAM and then starts monitoring the emulating code so that when it accesses the specific address at which the new value is to be loaded it can do so without effecting the emulation.

Emulator Comms - Interrupts

The emulator has the facility to generate an interrupt signal when a user definable area of memory is accessed. This facility will be useful for debugging purposes as this signal could then be used as a trigger in a logic analyser or an oscilloscope.

Three different IRQL Mode/Ranges are available:-

? BYTE: The target systems address must exactly match that specified by the software.

? 16 BYTES: Address lines A0 to A3 will be ignored. This means that if any location in a 16 address segment of the memory map is accessed then the interrupt will be generated.

? 256 BYTES: Address lines A0 to A7 are ignored. This gives a 256 address segment. Any access to an address within the 256 address segment will generate the interrupt.

To set up an interrupt you will need to specify the following information in the ***Emulator Parameters*** window:-

? IRQL Base address. This is the bottom address in memory for the interrupt range.

? IRQL Mode/Range. This specifies if want to generate an interrupt within 1 byte, 16 bytes or 256 bytes of the IRQL base address.

? Interrupt Pin Active state LOW or HIGH. This specifies that the interrupt signal should be either a logic high or a logic low.

The interrupt signal is programmable and can be set to active high or low. The signal is located on the red test probe on Emulator 1.

Show Me Help On Emulator Operations !

Emulator Comms - Example Source Code

Example Source Code

Here is an example of how to use the comms channel in your code when emulating EPROMs.
This version has been written in 8051.

Remember:- This code has been written using ANSI escape sequences which are not supported on WinLV.

```
      ORG 0000H

Comms_Base  EQU   1000H

Start:
      CALL  RESETCOMMS
      MOV   DPTR,#INIT_1      ; Send string init_1 to PC
      CALL  SEND_STRING
.1
      CALL  RECEIVE_BYTE      ; Receive data
  CALL      SEND_BYTE   ; & echo it back to PC
      JMP   .1

RESETCOMMS:
      MOV   R2,#0       ; wait more than 1ms
      DJNZ  R2,*
      DJNZ  R2,*
      DJNZ  R2,*
      RET

SEND_STRING:
      CLR   A
      MOVC  A,@A+DPTR   ;Read data from lookup table
      JZ    END_SEND_STR ;if rogue value then exit
      CALL  SEND_BYTE    ; send data to PC
      INC   DPTR        ; Ready to read next data
      JMP   SEND_STRING
END_SEND_STR:
      RET

SEND_BYTE:
      PUSH  DPH         ; Store data pointer
      PUSH  DPL
      MOV   DPTR,#COMMS_BASE  ; Initialise
  MOV B,A         ; store acc. in b
.1    MOV   A,#10H            ; Check for empty Buffer
      MOVC  A,@A+DPTR
      JB    ACC.1,.1
      MOV   A,B   ; Restore ACC to send low nibble
      ANL   A,#0FH
      MOVC  A,@A+DPTR ; Read BASE ADD+LOW NIB OF ACC
.2    MOV   a,#10H            ; Check for empty Buffer
      MOVC  A,@A+DPTR     ; Read status register
      JB    ACC.1,.2      ; Jmp if buffer full
      MOV   A,B   ; Restore ACC to send high nibble
      SWAP  A     ; get high nibble
      ANL   A,#0FH
      MOVC  A,@A+DPTR ; Read BASE ADD+LOW NIB OF ACC
  POP DPL      ; Restore data pointer
      POP   DPH
      RET

RECEIVE_BYTE:
      PUSH  DPH   ; store Data Pointer
      PUSH  DPL
      MOV   dptr,#Comms_Base ; Initialise dptr
.1    MOV   a,#10H            ; Wait for data to be sent
      MOVC  A,@A+DPTR   ; Read status register
      JNB   ACC.2,.1      ; jmp if no data in buffer
      MOV   A,#11H        ; Set address to COMMS_BASE+11h
      MOVC  A,@A+DPTR ; Read BASE ADD + LOW NIB OF A
      POP   DPL         ; Restore Data Pointer
      POP   DPH
      RET

      ; This text will appear on the screen when
  ; the comms window is opened
      ; ( N.B. 0dh prints as 0dh, 0ah )

INIT_1:
      DB 0dh,0dh,$1b,"[J",$1b,"[H",$1b,"[0;5;7;31m"
      DB    "ICE Technology LTD",$1b,"[0;1;36m",0dh
      DB    "Communications Example with ANSI escape sequences",0dh
      DB    "Press any key (ESC quits) :",0

      ORG   Comms_Base
      ; This area is reserved for use by the comms
      ; channel. This should stop you accidentally
  ; assembling code over the defined comms area
USED_AREA:
      DB    0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
      DB    0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
      end
```

Using Chiptest

What Does Chiptest Do?

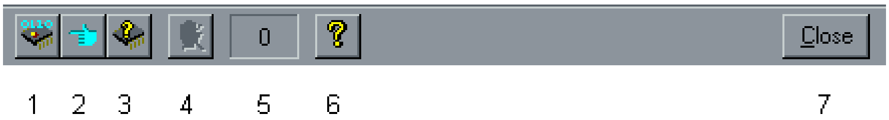
The WinLV Chiptest utility allows you to test certain devices to confirm that they are exactly what you expect them to be and also to check that the devices are functioning correctly. It does this be applying a combination of test vectors in order to determine exactly what the device is and to test its functionality. Chiptest can check the following device types:-

? 7400 Series Logic Parts
? 4000 Series Logic Parts
? Static RAMs
? Dynamic RAMs
? Complex Logic Parts
? User-Defined Logic Parts

Selecting a Chiptest Part

If you have a specific part to test and you already know its part number etc. you can select its library file using the Parts database . If you have a part that is unknown or it is not exactly clear then you can use the ***Match 74/4000 Device*** utility that will attempt to identify the part.

Programmer Operations Toolbar (Chiptest)



Programmer Operations Icons

1. Test Device. Click on this test the device in the ZIF socket.
2. Toggle Hands Free Mode. This allows you to select either – Test a device “Hands free” or Match 74/4000 Device “Hands free”.
3. Match 74/4000 Device. This operation will check the “unknown” device currently in the ZIF socket against all of the test vectors available (including any user-defined vectors) to determine what that device actually is.

   Show Me Help On Matching 74/4000 Devices !
4. “Hands Free Testing” (*Only Available on the 48 pin programmers*). This operation helps you to test or match large amounts of the same device. Once selected, follow the on screen instructions. Remember that you can toggle between Test and Match using ICON 2.
5. Chiptest counter display. Keeps a count of how many of the currently selected devices have been tested using Chiptest.
6. WinLV Help.
8. Close. Select this to close the ***Programmer Operations*** window.

Match 74/4000 Devices

What Does The *Match 74/4000 Devices* Utility Do ?

This utility will try to specifically identify any of the following parts by matching the libraries in the Parts database with the device to be tested:-

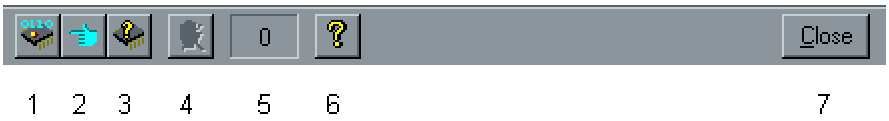
? 7400 Series Logic Parts

? 4000 Series Logic Parts

? User-Defined Logic Parts

This utility can be used when you have an unidentified device and you are unsure of its functionality. Once ***Match 74/4000 Device*** has found a match between the device being tested (placed in the ZIF socket) and an already existing library form the Parts database the result will be displayed in ***the Match 74/4000 Device*** dialogue box.

Programmer Operations Toolbar (Chiptest & Match 74/4000 Devices)



Programmer Operations Icons

1. Test Device. Click on this test the device in the ZIF socket.
2. Toggle Hands Free Mode. This allows you to select either – Test a device "Hands free" or Match 74/4000 Device "Hands free".
3. Match 74/4000 Device. This operation will check the "unknown" device currently in the ZIF socket against all of the test vectors available (including any user-defined vectors) to determine what that device actually is.
4. "Hands Free Testing" (*Only Available on the 48 pin programmers*). This operation helps you to test or match large amounts of the same device. Once selected, follow the on screen instructions. Remember that you can toggle between Test and Match using ICON 2.
5. Chiptest counter display. Keeps a count of how many of the currently selected devices have been tested using Chiptest.
6. WinLV Help.
7. Close. Select this to close the ***Programmer Operations*** window.

Edit Menu

The WinLV Edit Menu contains commands for easily manipulating the data displayed in the programming buffer. It Includes utilities for finding specific data in the buffer as well as byte/word swap and buffer fill and others.

Remember - The functions available in the Edit Menu will sometimes change depending on whether you are programming a device, using the Chiptest Utility or the ROM/RAM Emulator.

Copy (Ctrl+C)

This menu option allows highlighted sections of the programming buffer to be copied to a different location (address) in the buffer.

To copy a selection of addresses from one location in the buffer to another:-

? Move the mouse pointer to the start address of selection.
? Hold down the left mouse button whilst dragging the pointer across the buffer to include all address to be copied.
? Release the left mouse button when the mouse pointer is above the last address in the selection to be copied.
? Without touching the mouse buttons or pressing any other keys, select ***Copy*** from the ***Edit*** menu.
? Move the mouse pointer and click once on the start address in the buffer where you want to copy the highlighted selection.
? Select ***Paste*** from the ***Edit*** menu and the highlighted selection will be pasted into the new location in the buffer.

Paste (Ctrl+V)

This allows you to paste a copied, highlighted selection from one location in the buffer to another. (See ***Copy***)

Goto (Ctrl+G)

This allows you to locate and go to a specific address in the buffer. Enter the hexadecimal byte or word address in the ***Goto*** dialogue window and the buffer view will change to display the new location.

Find (Alt+F3)

This allows you to find a specific hexadecimal, octal or ASCII value in the currently open buffer. Select one the hexadecimal, octal or ASCII radio buttons and enter the value to search for in the ***Find String*** box.

I want to find a value in the buffer now !

Find next (F3)

After you have specified a search string using the ***Find*** menu option use ***find next*** to search through the buffer and locate the next occurrence of the search string.

Find previous (Shift+F3)

After you have specified a search string using the ***Find*** menu option use ***find previous*** to search back through the buffer , from currently selected address, and locate the previous occurrence of the search string.

Fill (Ctrl+F)

Allows you to fill the currently open buffer with a specified hexadecimal value. The ***Fill*** dialogue window allows you to enter a value (as a Byte or Word) and the start and end address within which all addresses will be filled with the selected value. The ***Fill*** default start and end addresses are set to the size of the currently open buffer.

I want to fill the buffer now !

Swap Bytes

Allows you to swap the MSB and LSB within any word in the currently open buffer. ***The Swap Bytes within Word*** dialogue window allows you to select the start and end addresses of an area of the buffer within which all word bytes will be swapped.

I want to swap bytes now !

Swap Words

Allows you to swap the Most Significant Word and Least Significant Word within any Long value in the currently open buffer. ***The Swap Words within Long*** dialogue window allows you to select the start and end addresses of an area of the buffer within which all words will be swapped.

I want to swap words now !

Checksum

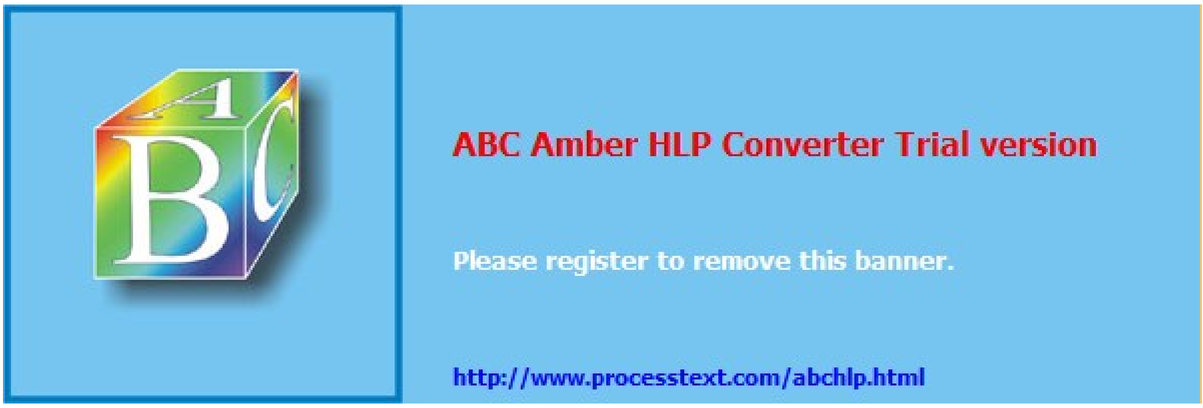
Allows you to calculate a buffer checksum (using the checksum calculator) or a device checksum . The checksum calculator can calculate a checksum for a user defined range of addresses in the buffer, or for the complete buffer. Checksums calculated are:-

? The sum of unsigned bytes with the carry being added to the MSB

? 1’s Complement (Inverse of the above checksum)

? CRC – Cyclic Redundancy Check

Show me the Checksum Calculator !



Programmer Menu (Gang Programming)

The WinLV Programmer Menu for Gang programming contains commands for controlling the operations of the GLV-32 programmer, selecting devices and setting up the stand alone mode .

Select Device

The ***Select Device*** menu item allows you to select which device you are currently using with your GLV-32 programmer. All physical devices - EPROMs & flash etc. can be selected from the parts database using the ***Select Device*** dialogue window.

Operations

Selecting the ***Operations*** menu item will open the ***Programmer Operations*** window so that you can control exactly what the GLV-32 programmer is doing.

Show Me Help On Programmer Operations For Gang Programming !

Set Mode

This allows you to select which programming mode the GLV-32 should use. The 3 different modes are:-

? ***Gang Mode***:-

Gang mode will set up the GLV-32 to programme all devices with identical data from the buffer, i.e.: 1 set of 8 identical EPROMs.

? ***4 sets of 2*** (32-bit mode):-

For 32-bit mode the programmer will send information into 4 sets of 2 chips. Sockets 0 and 1 will contain the LSB or the first byte specified from the buffer, and then every fourth byte. This maybe ODD or EVEN addresses, depending on the given Start address. Sockets 2 and 3 will contain the second byte and every fourth byte from there, sockets 4 and 5 will contain the third byte, and every fourth byte from there. Sockets 6 and 7 will contain the MSB or the fourth address, and then every fourth byte in the buffer.

? ***2 sets of 4*** (16-bit mode):-

For 16-bit mode the programmer will send information into 2 sets of 4 chips. Sockets 0 to 3 will contain the LSB or the first address specified from the buffer, and then every second byte in the buffer. This maybe ODD or EVEN addresses, depending on the given Start address. Sockets 4 to 7 will contain the MSB or the second specified address, and then every second byte in the buffer.

Set Up Internal EEPROM

This option is used to configure the internal EEPROM inside the GLV-32 programmer. This is necessary when you are using the GLV-32 in stand-alone mode Selecting this option will configure the GLV-32 to programme devices in stand-alone mode using the device currently selected in the software.

Show Me Help On Stand-Alone Mode !

Set Up Master EPROM

This option is used to configure the Master Configuration EPROM in the master socket on the GLV-32. This is necessary when you are using the GLV-32 in stand-alone mode

Show Me Help On Stand-Alone Mode !

GLV-32 Command Line Software

WinLV and the DOS software contains a utility that allows the Speedmaster GLV-32 Gang/Set Programmer to be operated using command line software at the DOS prompt. All GLV-32 functions featured in WinLV, such as programme, blankcheck, verify etc., can be invoked directly from the DOS prompt be using simple typed commands. The command line software provides the user with an alternative quick and efficient interface with the GLV-32 programmer which in turn can speed up production of large batches of the same device.

The command line software works by using a previously saved WinLV Project file (*.ice) and a collection of single word commands and command options to operate the GLV-32 hardware.

An application note is available from the website giving instructions on how to use the command line software.

Stand-Alone Mode

The Speedmaster GLV-32 gang programmer is capable of operating without being connected to a PC. This is known as stand-alone mode. The stand-alone mode allows the GLV-32 to be set up to programme a particular device with a specific data file so that it can be moved from one location to another without the need for a PC to control its actions.

Operating In Stand-Alone Mode

Once the configuration is complete, switch off the programmer. If you are using a Configuration EPROM, place the device in the Master socket. If you are using the Internal EEPROM, ensure that the Master socket is empty. When the GLV-32 is powered up, it waits to see if there is a message from the PC. If no message is received, it looks for a 27C512 EPROM in the Master socket. If no device is present, it looks in the internal EEPROM. If no data is found, or if there is a problem locating the Configuration EPROM, the Master LED will flash three times in succession and will continue this error message until the machine is reset.

Once the machine has found the correct device parameters, LEDs 0-7 will display a code byte indicating which type of device it is set up for. Once the code has been checked, clear the display by pressing reset.

Programming In Stand-Alone Mode

Place the DATA Master EPROM in the master socket, then place the blank devices in sockets 0-7. Press **START** to begin programming. While programming is taking place, the indicator LEDs will light for each socket in which there is a chip; the master LED will flash. During the VERIFY stage, the master LED will remain on permanently. If programming is successful, a “walking” sequence will appear on the LEDs (LEDs 0,4 1,5 2,6 3,7).. If unsuccessful, the LEDs will flash to indicate which sockets have failed.

Verifying In Stand-Alone Mode

Devices are automatically verified after programming. If it is wished to verify them separately, press and hold **RESET**, press **START** while still keeping **RESET** held down; then release **START**, then release **RESET**. For any device which fails to verify, the associated LED will flash. Press **RESET** to clear the error.

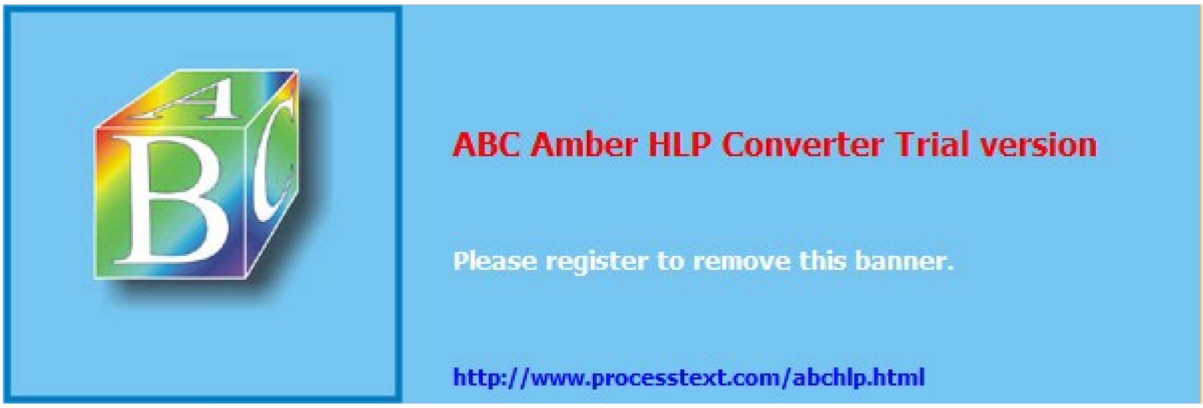
LED Power-Up Device Codes

What are Power-up Codes?

When the Speedmaster GLV-32 is set up for Stand Alone mode it will indicate which device the internal EEPROM is configured for using a power-up code. The power-up code operates using the LEDs numbered 0 to 7 . All device types have a Hexadecimal power-up code which is indicated as a binary sequence using the 8 LEDs. For example, if the internal EEPROM is configured for a 27C010 then the power-up code will be 07h indicated as binary 7 with LEDs 0,1 & 2 on and LEDs 3,4,5,6, & 7 off.

Example Power-up Codes

01	2716		02	2732		03	2764	
	04	27128		05	27256		06	27512
	07	27010		08	27020		09	27040
	0A	27080		etc				
	11	2816		etc				



LED Programmer Status Codes

Flashing Master LED repeated three times

BAD MASTER CHIP OR INTERNAL EEPROM NOT SET UP CORRECTLY

Flashing Master LED and any other flashing LED

MISALIGNED OR FAULTY DEVICE. IF ALL EIGHT SOCKET LEDs FLASH, THIS IMPLIES THAT ALL SOCKETS ARE FAULTY OR EMPTY.

Flashing Master LED and any other LEDs on

PROGRAMMING IS TAKING PLACE

Master LED on and all other LEDs off

VERIFYING MASTER EPROM

Master LED off and any other LED flashing

VERIFY ERROR

Master LED off and any sequence of LEDs on.

POWER UP CODE FOR DEVICE

Matrix
PROGRAMMING SYSTEM

The Matrix Programming System - an Introduction

The Matrix is a Universal Gang Programming System offering support for nearly 4000 devices including memory, logic and microcontrollers. It is available as either a 4 or 8 site unit. Each unit has a through port facility allowing multiple units to be daisy-chained together - allowing the control of up to 6 Matrix Systems from a single PC, that is up to 48 programming sites controlled by one user.

A “hands free” programming mode is available – allowing the user to instigate each programming operation with a single press of the button on the programmer it self after setting up a programming environment. A separate set of LEDS for each site on the programmer case report the status and result of programming of each device so the operative does not have to refer back to the PC once it is set up.

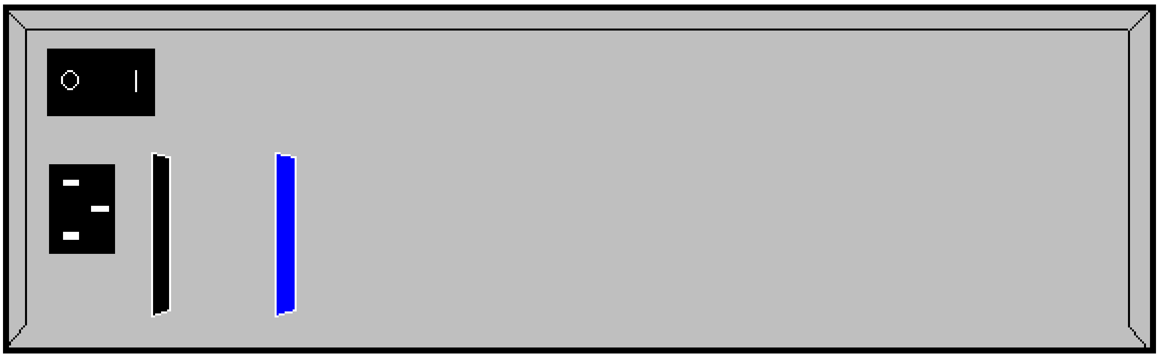
A full range of programming parameters and options are available to the user – allowing the programming environment to be tailored to individual preferences. This is explained in the section Matrix Programming Parameters .

The Matrix uses universal socket modules for all the most popular device packages up to 48 pins – pin counts above this and up to 100 pins are supported in more specialised modules. Modules can be purchased either individually or as sets of 4 or 8. Please contact Ice Technology Sales for more details.

The Matrix uses the same software as our other programmers and installation of its software is identical– with some additional features reflecting the additional facilities and features of this gang programmer. These differences are outlined in Matrix Programming Parameters. Connection is described in Connecting your Matrix .

Connecting your Matrix

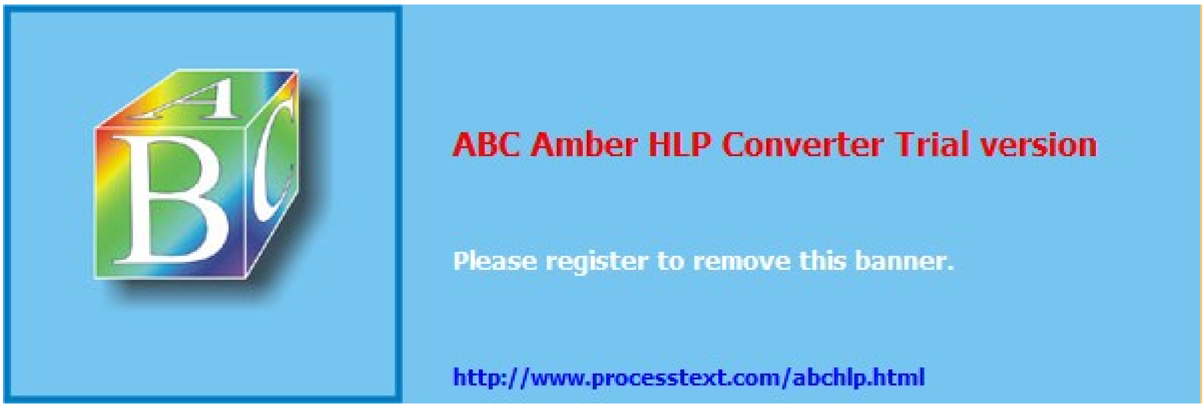
There are three connections available on the back of the Matrix as shown below:



The power connection is of the Universal IEC type. This is a universal input – the onboard power supply can accept any of the standard international Mains AC Voltages or frequencies.

The black D25 parallel connection is the IN port. When using a single Matrix, or if this is the first unit in a daisy-chained set of programmers, this connects directly to the LPT parallel connector on the PC controlling the programmer via the D25 parallel cable provided. When daisy-chaining multiple units this connects to the out port of the previous programmer.

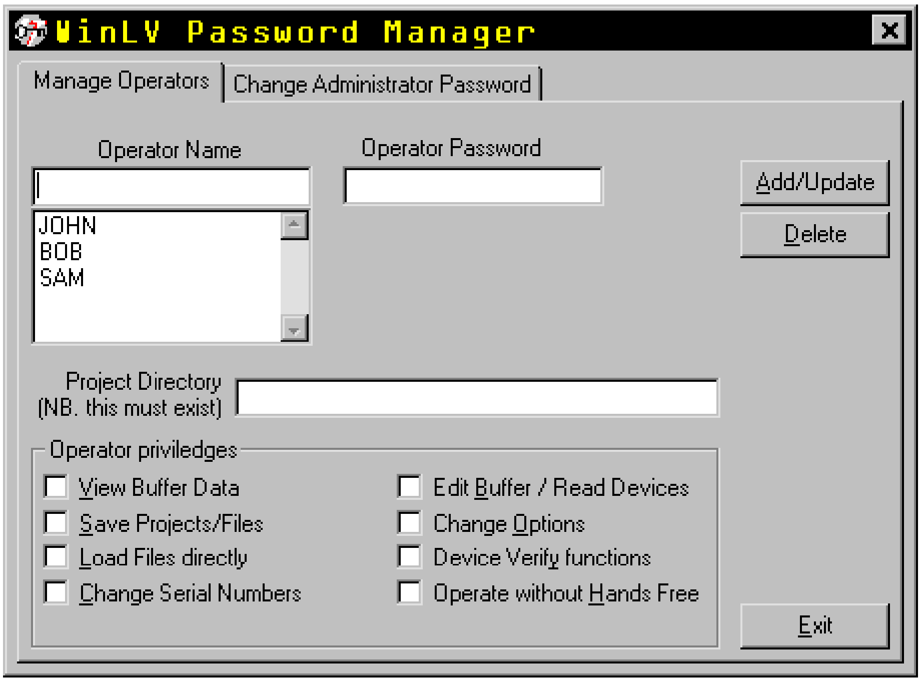
The blue D25 parallel connector is the OUT port – this will always be connected to the next programmer's IN port in a daisy chaining arrangement or not at all in the case of a single Matrix.



Setting Administrator/User Passwords and Privileges

When WinLV is first started, with either a Matrix connected or in Matrix demo mode, you will be asked for an administrator password. You can either leave this blank or enter the password you intend to use for administration/password purposes in the future.

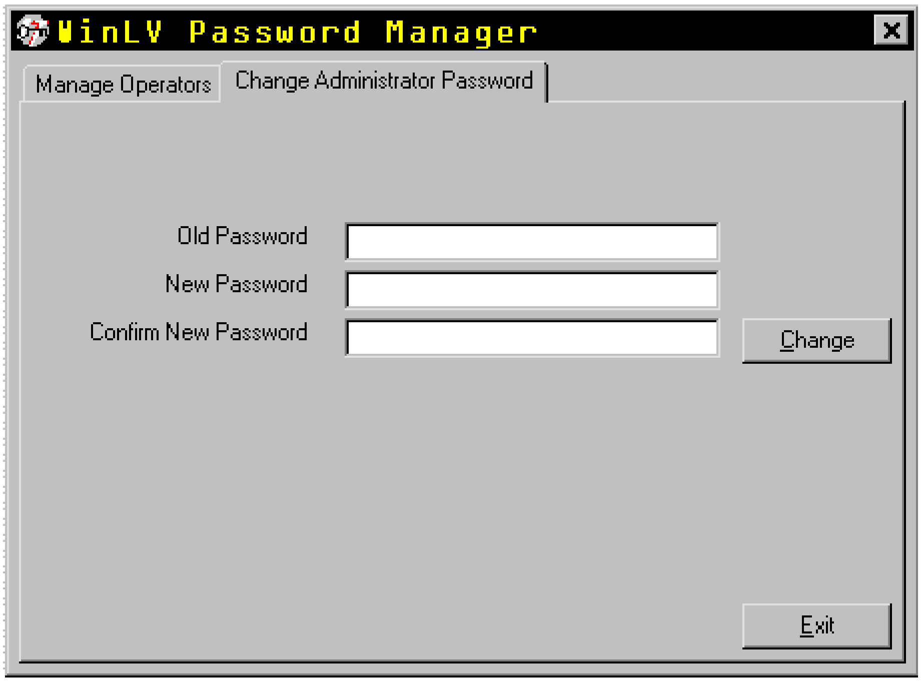
Once WinLV is running you will find an option in the main window, under file, called "Administrator Functions…". Clicking on this option will start the "WinLV Password Manager" application – with a window as shown below:



The Operator Name field allows you input the names of a user and the Password field must be used to allocate a unique password to that user. The box in the lower left hand side of the window contains descriptively labelled tick boxes that allow each user to be allocated with a very specific level of access. Each box can be ticked to allow the privilege described as desired.

The "Project Directory" field is used to define the path to which the given users log information is to be stored – the directory defined still has to actually exist and may need setting up in Windows.

Clicking on the tab labelled "Change Administrator Password" will open the window shown below:

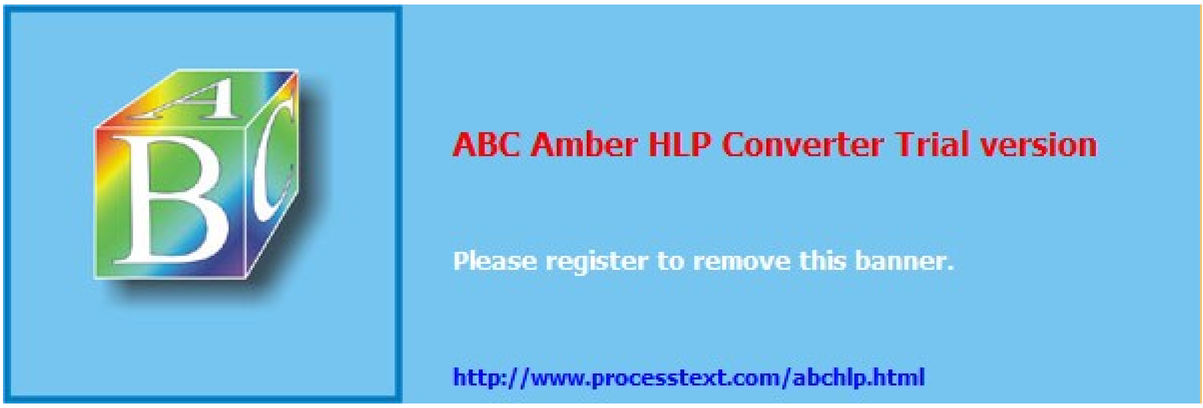


This allows the administrator to change the administration password.

ICE Technology Ltd. Contact Address

ICE Technology Ltd.
Penistone Court
Sheffield Road
Penistone
S YORKS S36 6HP
UK

Tel: +44 (0)1226 767404
Fax: +44 (0)1226 370434
email: sales@icetech.com



ICE Technology Ltd. Contact Address

ICE Technology Ltd.
Penistone Court
Sheffield Road
Penistone
S YORKS S36 6HP
UK

Tel: +44 (0)1226 767404
Fax: +44 (0)1226 370434
email: sales@icetech.com