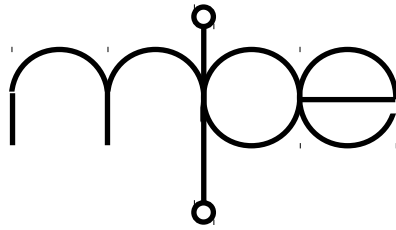# AIDE

Actually Integrated Development Environment

User Manual

AIDE

User manual

Manual revision 6.8

Date 02 September 2016

Software

Software version 6.8

Package Number:	

For technical support

Please contact your supplier

For further information

MicroProcessor Engineering Limited
133 Hill Lane, Southampton
SO15 5AF, UK
Tel: +44 (0)23 8063 1441
Fax: +44 (0)23 8033 9691

e-mail: mpe@mpeforth.com
          tech-support@mpeforth.com

web: www.mpeforth.com

Table of Contents

What is AIDE?

Pronounced "AIDE"

AIDE is a modular integrated development environment for the MPE series of tools and compiler. It provides a framework from which to use two types of tools.

1. MPE supplied compiler tools
2. External User tools
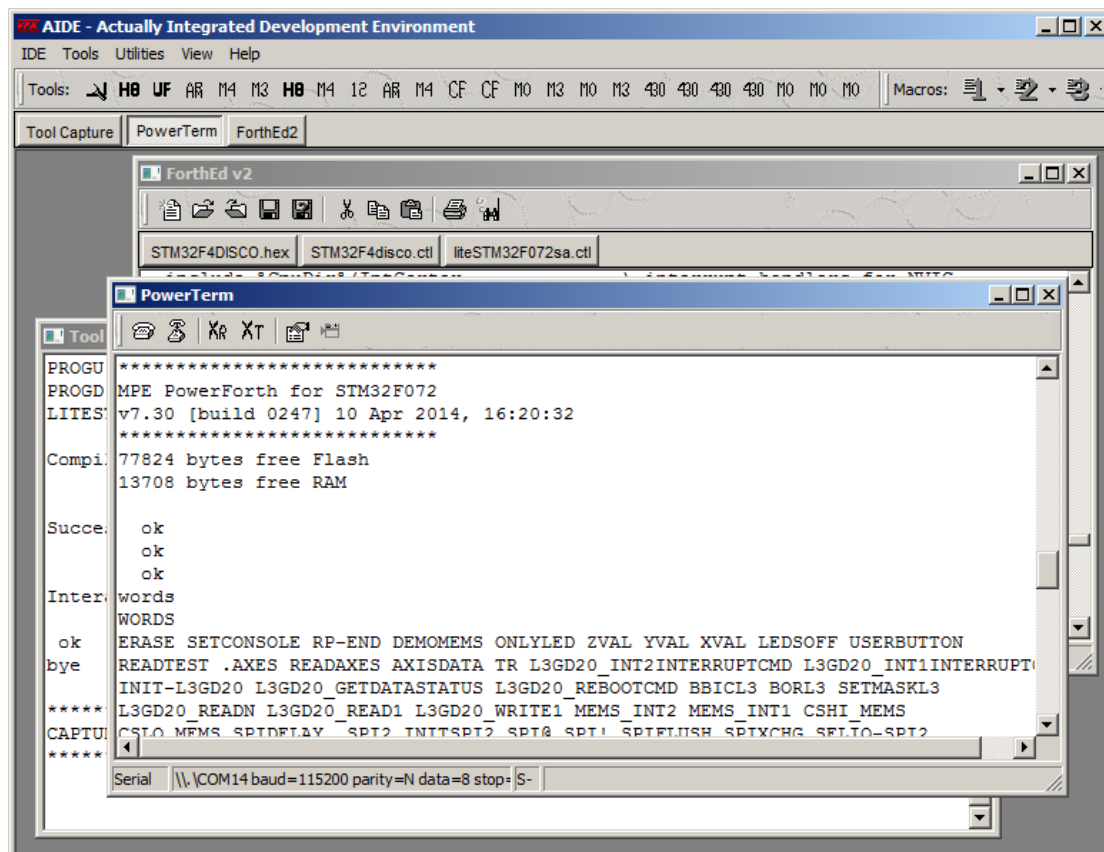
AIDE is written in Forth and the source code is available on request.

The Main IDE Frame

The main IDE consists of a single window frame with a "Tool Capture Display" Window (*See Capturing External Tools*).

The IDE provides the necessary functionality to:

- Launch both MPE and User supplied tools (*See External Tools*)
- Configure text substitution macros for use in command lines to launch tools.
- Automatically include and maintain "plugin" tools.

The Main Menu Bar
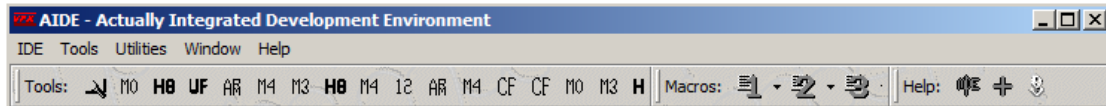
The *IDE* Menu

This menu provides options for general configuration of the IDE. Available sub-options are:

Configure

Setup general IDE options, especially the editor and **LOCATE** facilities. Within the editor configuration string 'f' will be replaced by the file name and 'l' will be replaced by the line number, for example:

```
CodeWright     "%f%" -g%l%
Crimson        /L:%l% "%f%"
Ed4Windows     -1 -n -l %l% "%f%"
               N.B. the first is minus 1
EMACS          --no-wait +%l% "%f%"
PFE32          /G%L% "%f%"
Programmer's Notepad
               -l %l% "%f%"
UltraEdit      "%f%" -l%l%
WinEdit        "%f%" -# %l%
```

Editor path names containing spaces must be entered surrounded by quotes in the IDE->Configure dialog box. We have worked around some problems with the Windows API when saving these path names. When entering the locate command string in this dialog you must put quotes around the **%f%** macro in order to handle filenames containing spaces. If you do this, you must also specify the line macro **%l%** to prevent Windows from removing the quote marks. The following (for UltraEdit) works:
`"%f%" -l%l%`
but
`"%f%"`
will eventually fail.

To help you configure **LOCATE**, there's a checkbox to produce debug information. The display format is crude, so please remember to to uncheck this when **LOCATE** is working properly.

Macro Management

Alter the names and alternatives for the built in text macros.

External Tools

Configure and manage external tool options.

2

Save INI file as …
> The working INI file is saved . You have to enter a file name for the saved file.

Load INI file from …
> Enter a file name, and when AIDE is next restarted, the given file is copied to the working INI file.

Exit
> **C**loses AIDE.

The *Tools* Menu

This menu contains a list of the currently configured external tools. Selecting an item here will run the specified tool. This menu is also reflected as a toolbar within the IDE.

The *Utilities* Menu

This menu contains internal tools. Selecting an item here will run the specified tool. The default options are:

Calculator
> Run the usual Windows calculator.

ASCII Chart
> Display the ASCII table in hex, decimal, and control code descriptions.

BMP to LCD Converter
> Convert picture in .BMP format to a native LCD format.

Tool Capture
> If ticked, the tool capture window is open.

PowerTerm
> If ticked, the PowerTerm window is open. You can toggle this.

ForthEd2
> If ticked, the ForthEd2 window is open. You can toggle this.

Forth Console
> If clicked, a Forth console opens.

Go Online
> Launch your web browser.

The *Help* Menu

This menu provides various information and help options such as MPE contact details, product version information and manuals.

The Main Window "CoolBar"

In addition to the main window menu the IDE frame also provides three toolbars organized in a "coolbar" format. Two of these toolbars echo the functionality and options of the **Tools** and the **Help** sub-menus. The third toolbar provides control for the text substitution macros. (*See Text Substitution Macros*)

Configuring and Maintaining External Tools

About External Tools

An *External Tool* is simply another program outside AIDE which is invoked in a specified manner from within AIDE by clicking a button on the toolbar or from the *Tools* menu. These tools can be DOS tools, Console Mode apps or Windows GUI applications. Simple "fire-and-forget" tools such as MAKE can have their input and output revectored to an IDE window rather than a separate console. There is a limit of 26 tools which can be added to the IDE. The tool management dialog is invoked by the **"*External Tools*"** option on the *IDE* menu.

You **must** use the *Save* button before the *Exit* button when you change an item in the dialog..

IDE Parameters

These are the options which control how the tool is represented within the main IDE Frame.

Tool Name

The name is the tool name as it will appear on the *TOOL* menu and tooltips.

Tool Bitmap

A path to the image file to use on the coolbar to represent the tool. A *Browse* button is supplied for locating bitmap images. The bitmap must be 16 color (Windows) and must be 16*16 pixels in size. If you want to create buttons yourself, you can do this with the Paint program, *mspaint.exe*.

Help Item checkbox

If this box is checked, the tool will appear on the Help menu. Note that any program can be launched this way, and so this is a useful way of accessing project information such as chip data sheets which are normally provided as PDF files. See "*Command Line*" below for more details.

Tool Execution Parameters

These parameters control how the tool is to be launched.

Program/File

This box contains the command line for executing the tool. This can be either a full path or a path relative to the *Start Directory* as defined below.

Command Tail

Command Line options can be added and text macros can be used to form part or all of the command line.

If your tool has a space in the path, e.g. it is somewhere in the *Program Files* folder, you should separate the program name and keep it here. Put the rest of the command line in the "*Command tai*l" box.

If you just want to display a web page or a PDF file, and you have an association set up (you do by default), just put the full pathname here, leave "*Command tail*" box empty and put a single dot in the *"Start Directory"* box

Start Directory

This represents a full path which will form the current working directory when the tool is launched.

Captured Tool Options

These parameters control input/output redirection of command line tools to the IDE.

Keyboard Strokes

Only valid for a captured tool. Any text in this box is passed to STDIN of the tool after it is launched.

Capture Output To IDE

When checked, all console IO for this tool will be redirected to the IDE capture window.

Notify on finish

When checked, a banner is displayed in the capture window when the tool terminates.

Act on Commands

Enables AIDE to react to AIDE-specific commands from the tool. The MPE Forth Cross Compilers use this feature to launch the editor and to display the source for Forth words.

Clear screen

The IDE capture window will be cleared before the tool is run.

Other Buttons

Prev

Cycle display to the previous external tool.

Next

Cycle display to the next defined external tool.

Add Tool

Add a new blank tool entry. The options can then be filled in.

Remove Tool

Delete the current tool entry. Note that the last tool in the list (Explorer) cannot be removed.

Copy Tool

Create a new tool entry and fill its configuration with that of the previous current tool.

Exit

Leave this dialog.

Save

Apply changes to current tool. NOTE: Changes are not committed any other way. If you fail to *Save* changes before using Add,Next etc. **they will be forgotten**.

Upgrading from previous version of AIDE

If you are upgrading from a previous version of AIDE which did not have the *"Command Tail"* entry, you will have to update your configurations manually. Don't forget to press *"Save"* for each one. Where you previously had a *"Command Line"* entry:

```
c:\buildkit.dev\software\compiler\xmsp430.exe /ide /pauseoff
```

split this into the two entries *"Program/File"* and *"Command tail"*

```
c:\buildkit.dev\software\compiler\xmsp430.exe
/ide /pauseoff
```

If you are also upgrading to a version 7 Forth cross compiler, the Command Tail entry should be changed to:

```
-ide /ide /pauseoff
```

where the `-ide` switch causes the VFX Forth host to configure itself for AIDE, and the `/ide` switch causes the cross compiler to use AIDE to launch the editor for **LOCATE** commands.

Merging INI files

AIDE saves its configuration in a file called A*ide.ini*. This file is kept in a directory called *%AppData%\MPE\Aide*, where *%AppData%* is operating system and user specific, for example for one MPE box, *%AppData%* expands to *C:\Users\Stephen\AppData\Local*. A side effect of this is that Aide.ini is global to all instances of AIDE. See below for using separate configuration files.

This can cause trouble when you install a new cross-compiler. Because there is already a file called *%AppData%\MPE\Aide*, a new one is not copied from the directory containing AIDE. However, it is not difficult to extract the external tool information from the new *Aide.ini* and add it to the existing one. There are two approaches to performing the merge. One is to print out the new INI file and manually enter the data using the IDE->External Tools Dialog. The other is to edit the old INI file directly.

Editing *Aide.ini*

Shutdown all copies of AIDE.

Find the new *aide.exe* and you will find another *Aide.ini* in the same directory.

In the new *Aide.ini,* copy the text describing the external compilers. These are of the form:
```
eaaadwtype                  = 2
eaaasztoolname              = STM32-P107
eaaaszbitmapcolor           =
C:\buildkit.dev\software\aide\CM3.bmp
eaaaidcommand               = 58720
eaaaszcommandline           =
c:\buildkit.dev\software\compiler\xArmCortexDev.exe
eaaaszcommandtail           = -ide /ide /pauseoff
```

```
eaaaszstartdirectory         =
C:\buildkit.dev\software\ROM\Cortex\Hardware\STM32
eaaaszkeystroke              = include STM32-P107.ctl
eaaabcaptured                = -1
eaaadwactionontermination    = -1
eaaapreclear                 = -1
eaaabhelpitem                = 0
eaaabactcommand              = -1
```
Each line corresponds to an entry in the IDE->External Tools dialog.

Each tool description line starts with Xaaa, where X runs from A..Z and must be contiguous. Now place all the tools you want in *%AppData%\MPE\Aide\Aide.ini*, Edit the first letter of each tool line so that each tool has a unique character and these characters are contiguous.

Save the file and restart AIDE.

Using separate configuration files

When you have several copies of AIDE and want each one to use its own INI file, launch the secondary copies of AIDE with the *-ini-exec* command line switch, e.g.
  C:\buildkit.dev\software\aide.dev\aide.exe -ini-exec
and AIDE will run with the INI file in the directory containing *aide.exe,* in this case
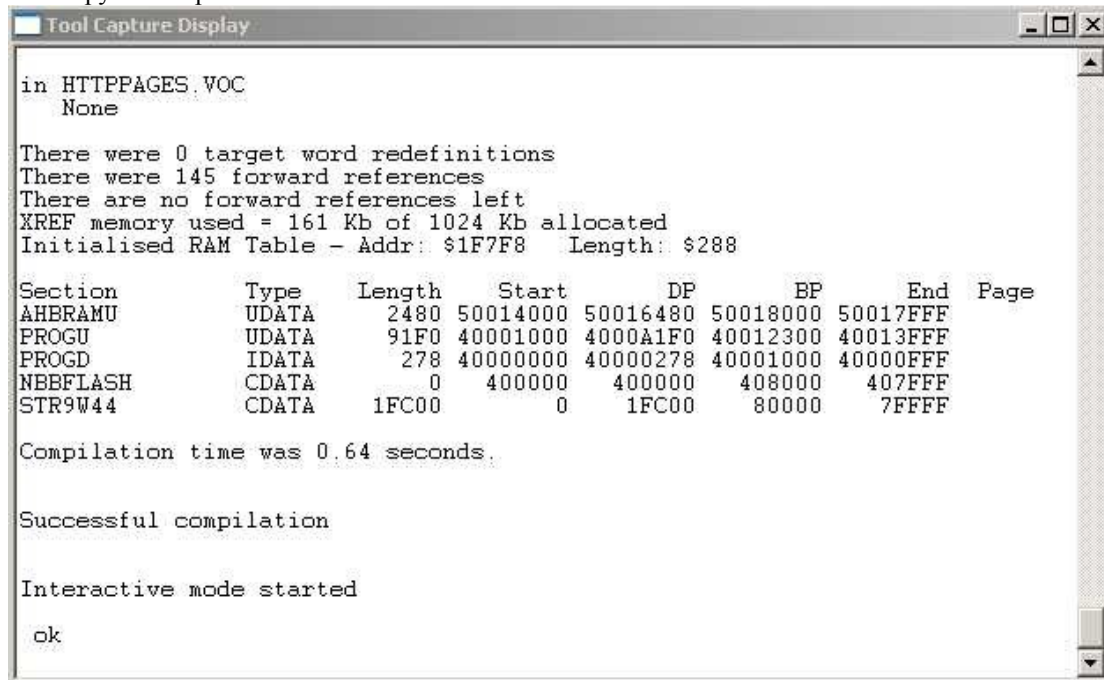 C:\buildkit.dev\software\aide.dev
The switch *-ini-exec* must be in lower case.

This arrangement is particularly useful when you are evaluating compilers such as the Lite versions and do not want to have to merge INI files.

The Tool Capture Display Window

This window is a part of the IDE framework by default. It provides a console, which can act as the IO device for a captured external tool. (See previous section). This window handles the Windows standard cut/copy and paste keyboard shortcuts. It has a right-click context menu.

Pressing the right hand mouse button within the capture window will pop-up a menu giving Cut/Copy/Paste options as well as Print and Text Search facilities.

```
Tool Capture Display                                                    _ □ ×

in HTTPPAGES.VOC
    None

There were 0 target word redefinitions
There were 145 forward references
There are no forward references left
XREF memory used = 161 Kb of 1024 Kb allocated
Initialised RAM Table — Addr: $1F7F8   Length: $288

Section          Type    Length    Start       DP        BP       End   Page
AHBRAMU          UDATA     2480 50014000 50016480 50018000 50017FFF
PROGU            UDATA      91F0 40001000 4000A1F0 40012300 40013FFF
PROGD            IDATA       278 40000000 40000278 40001000 40000FFF
NBBFLASH         CDATA         0   400000   400000   408000   407FFF
STR9W44          CDATA     1FC00        0     1FC00    80000    7FFFF

Compilation time was 0.64 seconds.


Successful compilation


Interactive mode started

 ok
```

Text Macro Substitutions

About Text Macros

All strings used in the configuration of the IDE and the External tools can have text macro expansion. The IDE provides eight user definable text macros, each of which can have up to eight different translations depending on user selection.

Text macros are used by placing their names (surrounded by percentage character %) within a text string. For example, a text-macro called *FRED* could form the path to a command line using:

You can define two different versions of this macro, one for use on your desktop, and another for use on your laptop. The availability of these macros allows you to copy folders from desktop to laptop without having to edit source code files.

Note that the MPE Forth cross compilers also support text macros, but the macros within the cross compiler are separate from those used by AIDE. All AIDE macros are expanded (substituted) before the string to passed to the compiler or other external tool.

Changing Macro Names

The IDE provides 8 text macros. The names for these macros can be edited using the dialog provided by the menu option *Macro Management* on the **IDE** sub-menu.
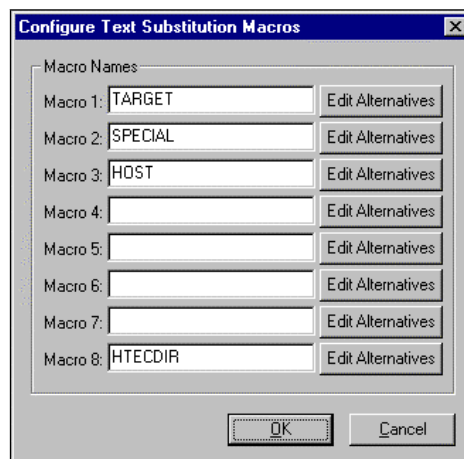
This dialog allows you to change the name of each text macro.

Editing Macro Alternatives

The **Edit Alternatives** buttons allow you to configure the 8 possible alternatives to the translation of each macro.
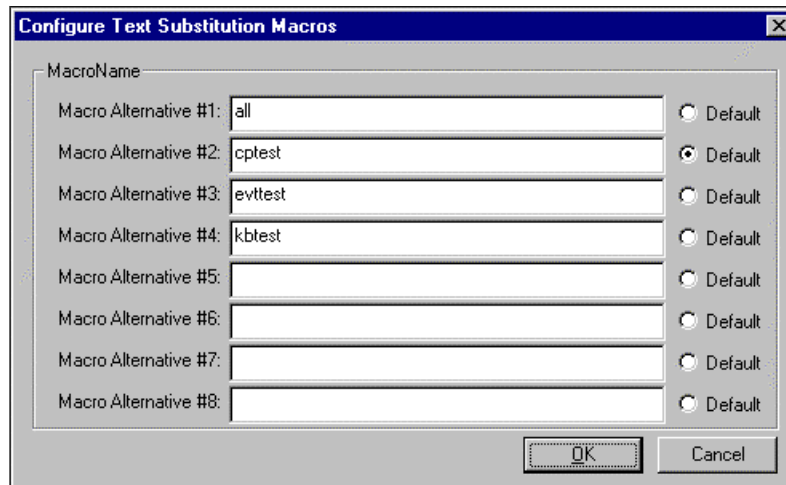
Selecting Active Macro Translation

Since each of the 8 text macros can have up to 8 possible alternative translations the IDE needs to provide a method of switching translations. This can be achieved by entering the **Configure Text Substitution Macros** dialog as before but can also be done via the main IDE "coolbar".

On the main coolbar each macro has a button which can drop-down a menu. The dropdown menu lists the possible alternatives and places a tick next to the currently selected translation. Simply click on the required translation to change.
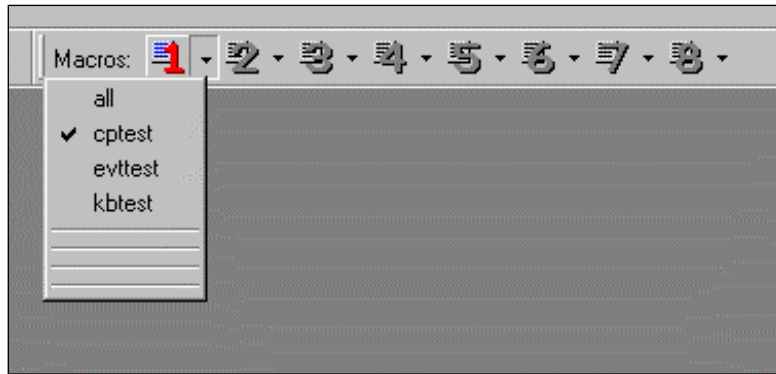
PowerTerm

About PowerTerm

PowerTerm is an RS232 Terminal window for AIDE.

The PowerTerm window has 3 major parts.
1. The toolbar. This provides the Connect/Disconnect, Configure and Logging options
2. The Main Terminal display Window
3. The Status display.
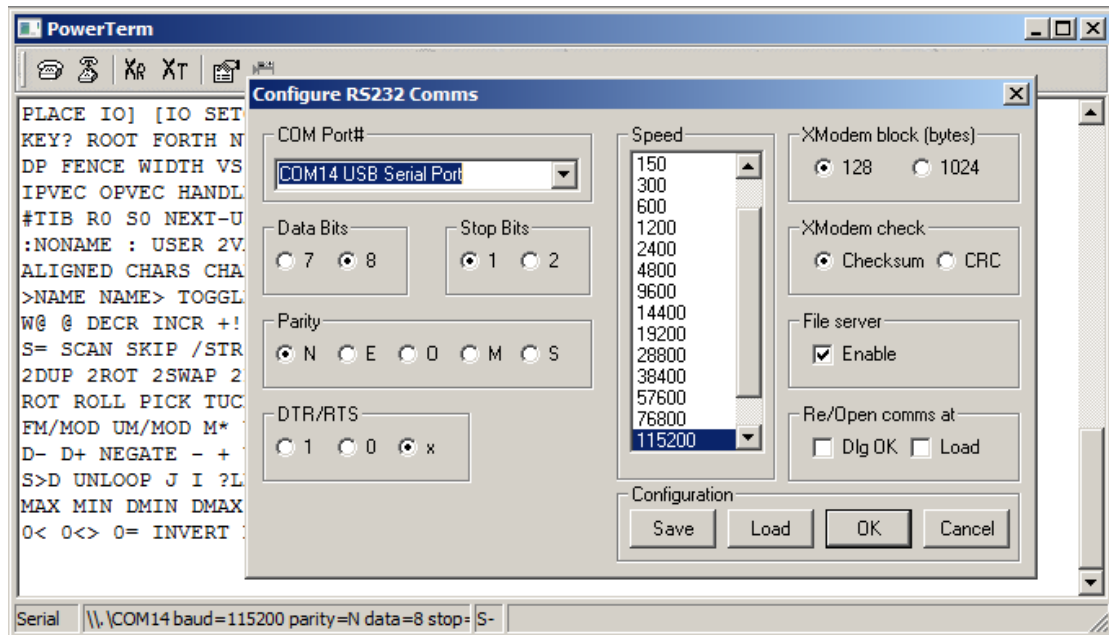
You can get to most of the functions of PowerTerm from a right-click menu.

Toolbar Buttons

Connect

This option will attempt to open the configured serial port.

Disconnect

Close any currently open serial port.
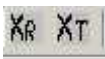
Xmodem Receive/Transmit

These two buttons allow you to receive and transmit binary file in Xmodem format. Xmodem configuration defaults to 128 bytes and checksum. You can change this in the Server and XMODEM panel of the configuration dialog. Transfers using 1024 byte blocks with CRCs are much faster, especially with USB serial ports.

Configure

Change the current configuration of the serial port, Xmodem and the console window. Many USB serial devices need high COM port numbers. You can find out

which is which using *Control Panel -> System -> Hardware -> Device Manager -> Ports* or a similar incantation. Recent versions of PowerTerm show the "friendly" device names to make life easier for you

Console window configuration includes enabling and disabling of the file server facilities, which can be used by target hardware using the MPE ROM PowerForth code (usually in the COMMON\ROMFORTH directory). These facilities are documented in the ROM PowerForth manual.

Log console to file

This is a toggle that controls logging of the console to a file. When logging is enabled, you will be presented with a file selector dialog to select the file for logging. This toggle can also be activated by pressing ^T.

The Status Bar

The first panel indicates the connection status - "Serial" or "Closed".

The second panel shows the connection conditions, e.g. " COM6: baud=115200 parity=N data=8 stop=1".

The third panel contains 'S' if the file server is enabled and 'L' if data is looged to a file.

The fourth panel contains a line number during a file include or a block number during an Xmodem transfer.

Shortcut keys

The following shortcut keys are available:
^C – Copy highlighted text to the clipboard
^V – Paste text from the C[ipboard (also Shift-Insert).
^K – send time and date to the target as a Forth string. This is useful to set the real time clock of a target board. The format of the string is:
```
%d to dow  %d %d %d %d %d %d SetTime&Date
```
where %d corresponds to a literal number, **dow** is a Forth **VALUE** containing the day of the week and the parameters to **SetTime&Date** are as for **Time&Date**.
^L – clear the display.
^T – start/stop logging.
^Y – start XModem receiver.
^Z – start XModem-128 transmitter
^\ - start XModem-1024 transmitter

Auxiliary debug displays

The main PowerTerm console is not cursor-addressable, it was designed for logging data. For those occasions when you need an old-fashioned display, or when you need multiple displays for a multi-tasked application, you can use the auxiliary debug displays. These are traditional cursor-addressable displays.

The displays are controlled by sending control sequences starting with <ESC><!>. Unless otherwise stated parameters are single bytes in the range 0..255. The whole of the command sequence must be performed with inter-character gaps less than 100 ms. For examples of using these commands see the target files *Common\DebugChan.fth* and *Examples\hanoi.fth*.

```
<ESC>!'0'       Use the main PowerTerm console – char code 0
<ESC>!'1'       Use the the first debug display – char code 1
<ESC>!'2'       Use the the second debug display – char code 2
<ESC>!...
<ESC>!'8'       Use the the eighth debug display – char code 8
```

```
                If a display is not open, it is created.
                If it has not been defined, the default is
                80x25 in a 10 point font.
<ESC>!'9'n   Close display n (1..8).
<ESC>!'10'nwhf Define the display size and open it
                n is the display in the range 1..8
                w (width) and h (height) are 1..255.
                f (font height) is 6..24
                The display is not selected for output.
<ESC>!'11'n#<txt> Set the caption of an open display
                n is the display, 1..8
                # is the number of characters in the string
                txt is the string (no terminator)
<ESC>!'12'nfffbbb Set the foreground and background colours
                n is the display in the range 1..8
                fff is a 24 bit foreground colour (big-endian)
                bbb is a 24 bit background colour (big-endian)
<ESC>!'13'n   Clear the display to the background colour
                n is the display in the range 1..8
<ESC>!'14'nxy   Goto the given position
                n is the display in the range 1..8
                x is the X position from the left
                y is the Y position from the top
```

To use these facilities, compile the file *Common\DebugChan.fth* into your target to provide a set of words for accessing the displays. An example using the old Towers of Hanoi as a demonstration can be found in *Examples\hanoi.fth*.
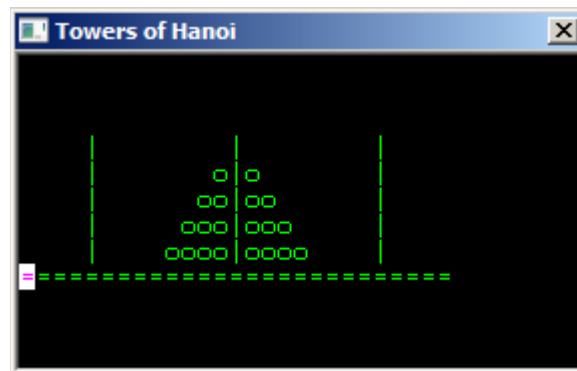


### Gotchas

The target should be set up to send the CR character (ASCII code 13) before the LF character (ASCII code 10) for correct display. Unfortunately Windows does not recognise LF before CR as a valid sequence, leading to the display of a curious character instead.

When the file server is enabled, some control characters will trigger the internal file server. If you are debugging a target system that is prone to crash or produce erratic characters, you should disable the file server.

Note that when using USB serial adaptors, especially on laptops, you may need to use *Control Panel -> System -> Device Manager* to find out which COM port number is to be used. However, PowerTerm's configuration dialog does a good job of listing the 'friendly' names of the COM ports.

Some USB serial adapters, especially generic ones, have simply dreadful Windows device drivers. Always install the latest drivers for your specific device. We generally buy FTDI cables directly from FTDI. We have also had good results with Keyspan devices. If things go wrong with adapters based on the Prolific chip sets you can always try downloading the latest drivers from Prolific. Assuming that the

latest Windows drivers are up to date is a mistake. Yes, we are tired of performing technical support for serial issues caused by bad device drivers.
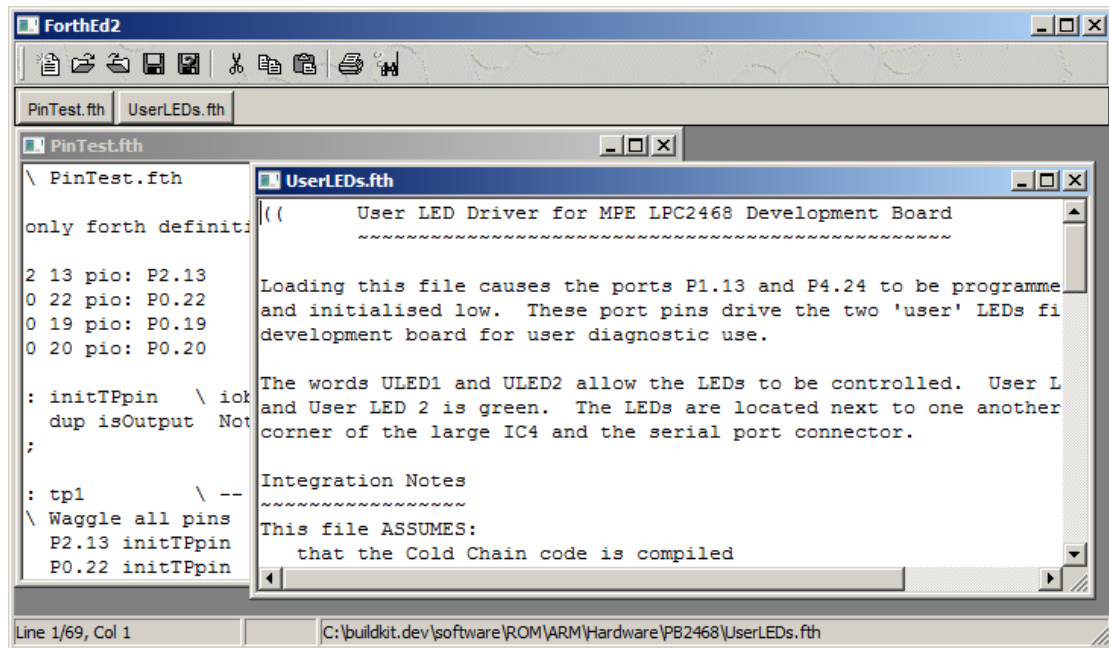
# ForthEd2

## About ForthEd2

ForthEd2 is a simple ASCII text editor for AIDE. It allows the simultaneous editing of up to 16 files with full text search, clipboard access and printer support.

ForthEd2 is also linked to remote commands from the tool capture window. If you have enabled the "Act on command channel" checkbox in the external tool configuration, the MPE cross compiler **LOCATE** command will show the source code of the word in ForthEd.



You can get to most of the functions of ForthEd2 from a right-click menu.

Note that ForthEd2 is **not** a production-grade programmer's editor. It's provided because IDEs have to have an editor, so AIDE has one! There are plenty of really good programmers editors available at very low cost. MPE's current favourite is *UltradEdit* from *www.ultraedit.com*.

## Display/Edit Window

The editors main display window consists of a tabbed control area. Each file currently open has a tab, which you can use to navigate from file to file.

## Toolbar Buttons

### New File
Create a new blank file and select it.

### Open File
Create a new tab window and load in the contents of a given text file.

### Close File
Close the currently active text file. This option does **not** perform a *Save*.

### Save File
Save the current text file. The name to be used for the destination file is the same as the tab control title. The file is **not** closed.

### Save File As
Save the current text file. The editor will ask for a filename.

### Cut
Cut the selected text into the Windows clipboard.

### Copy
Copy the selected text to the Windows clipboard.

### Paste
Paste text from the Windows clipboard to the current cursor position.

### Print
Print the current file.

### Search
Search the currrent file for text strings.


## Shortcut keys

The following shortcut keys are available while editing:
^S - save
^C - copy to clipboard
^X - copy to clipboard and delete
^V - paste from clipboard (also Shift-Insert)
^F - Find/Search
^N - New
^O - Open
^P - Print

# MPE Forth Cross-Compiler and AIDE

The MPE Cross-Compilers have built-in support for the AIDE development environment.

## Command line switches

By default, an MPE cross-compiler is a standalone Windows GUI program. There are switches which can be appended to the cross-compiler command line in order to run it as a captured external tool from within AIDE.

### Forth 7

The relevant switches are:

| | |
|---|---|
| **-IDE** | Run compiler as a Win32 Console Application. Must be the first item on the command tail |
| **/PAUSEOFF** | Close down compiler when compilation complete. |
| **/IDE** | Use AIDE for **LOCATE**. |

### Forth 6

The relevant switches are:

| | |
|---|---|
| **/IDE** | Run compiler as a Win32 Console Application. Must be the first entry on the command tail) |
| **/PAUSEOFF** | Close down compiler when compilation complete. |

## ForthEd2

ForthEd2 can be linked to remote commands from the compiler. If you have enabled the *"Act on command channel"* checkbox in the external tool configuration, the cross compiler **LOCATE** command will show the source code of the word in ForthEd2. Setting the **INTERACTIVE** and **+XREFS** switches in your control file, and then using the **XREF <name>** and **LOCATE <name>** facilities provides very powerful and quick source management facilities.

## Umbilical Forth Serial Comms

When using the Umbilical Forth system you should not use *PowerTerm* to access the target board. If the terminal is active the live-Forth in the cross-compiler cannot use that serial port to make the umbilical link. **Disconnect PowerTerm before using Umbilical Forth.**

## Stand-alone Forths

PowerTerm can be used with all MPE stand-alone Forths.

## ROM PowerForth

PowerTerm includes facilities that allow targets built by the MPE Forth6 and VFX cross compilers to compile source code directly from the host PC. These are described fully in the ROM PowerForth manual. Note that the Forth6 target ROMFORTH code is **not** compatible with the code for the earlier XShell front ends.

```
UP-LOAD   (--)
```
PowerTerm will ask you which file you wish to send, and the file will then be compiled. Note that because this protocol also works with other terminal emulators, the last line in the file **must** include the word **END-UP-LOAD**.

```
GET <filename>   ("<filename>" --)
INCLUDE <filename>   ("<filename>" --)
```
Running these on the target triggers the file server to download source code which is compiled by the target. Files may be nested and require no special terminating condition.

```
HEX-DOWN   (addr len --)
```
Transmits an intel hex file to the host. PowerTerm can capture this using the logging function.

```
BIN-DOWN   (addr len --)
```
Triggers the XMODEM receiver in PowerTerm.  The name of the file to be created will asked for by PowerTerm.

## Custom versions of AIDE

AIDE is written in Forth, specifically for VFX Forth for Windows.

If you want to customise AIDE yourself, you can get the source code for AIDE by obtaining a Developer edition of a cross compiler. Then ask MPE for the source code if it was not provided with the cross compiler. You may have to sign a Non-Disclosure Agreement (NDA).

# Index