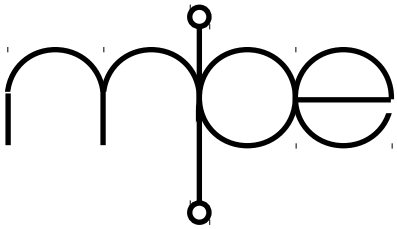

MICROPROCESSOR ENGINEERING LIMITED

133 Hill Lane, Southampton SO15 5AF, England
Tel: +44 (0)23 8063 1441 Fax +44 (0)23 8033 9691
email: tech-support@mpeforth.com
mpe@mpeforth.com
14 November 2013

Getting started with MPE Forth Cross Compilers

Installation

Windows

Run the setup file you have downloaded. It will install all the software for you. You may need to provide the unlock key when prompted for it. This is a 12 digit number and should be entered without spaces or other punctuation. The key is usually found in the email with the download instructions or on the invoice.

An uninstaller is provided and is available in the program group after installation.

After installation you will find the following directory structure inside the main directory:

```
Xcpu
  Cpu               CPU specific target code
    Configs         Hardware control files
    Drivers         Peripheral drivers
    Hardware        Board specific code
  Aide              The compiler front end IDE
  Common            Common high level target code
    RomForth        ROM PowerForth extension
    Tests           MPE and other test suites
  Compiler          Forth cross compiler
    Source          Source code if supplied
    Xtra            Windows NT/2000/XP port driver
  Docs              Documentation and manuals
    ProgRef         Chip specific documentation
    ANSForth        ANS Forth standard
  Examples          Sample target code
  Tools             Miscellaneous tools
```

Note that the Forth Stamp versions may not include all the directories. The Stamp versions are code and data space limited and do not include :

- Filing system, PID controller and State Machine compiler from the Examples directory,
- Docs\ProgRef directory,
- Tools directory except for Hex/S19 file converters.

The cross compiler itself is in the `Compiler` folder and is usually called `x<cpu><type>.exe` where type is `Dev` or `Stamp`.

Now please read at least the installation and configuration chapter of `Docs/XC7man.pdf`. We do document our products.

Mac OS X

The code is supplied as a zipped tarball, usually called **x<cpu><ver>OSX.tar.gz**. For example, the Developer version of the ARM toolchain is called **`xArmDevOSX.tar.gz`**. To install it (say) in your home directory, open a Terminal and switch to your home directory as the current directory – Terminal usually starts in your home directory.

```
$ cd ~
```

and run:

```
$ tar -xvvzf xArmDevOSX.tar.gz
```

Now you have to put the executables wherever you need them. Switch to the directory you have just created:

```
$ cd xArmDev
```

There you will find a shell script file **`InstallMe.osx.sh`**. Check and perhaps edit this file. You may have to use **`sudo`**.

```
$ ./InstallMe.Osx.sh
```

or

```
$ sudo ./InstallMe.Osx sh
```

The cross compiler itself is in the `Compiler` directory and is usually called

```
x<cpu><type>OSX
```

where type is `Dev` or `Stamp`.

Now please read at least the installation and configuration chapter of `Docs/XC7man.pdf`. We do document our products.

Linux

The code is supplied as a zipped tarball, usually called **x<cpu><ver>Lin.tar.gz**. For example, the Developer version of the ARM toolchain is called **`xArmDevLin.tar.gz`**. To install it (say) in your home directory, open a Terminal and switch to your home directory as the current directory – Terminal usually starts in your home directory.

```
$ cd ~
```

and run:

```
$ tar -xvvzf xArmDev.tar.gz
```

Now you have to put the executables wherever you need them. Switch to the directory you have just created:

```
$ cd xArmDev
```

There you will find a shell script file **`InstallMe.lin.sh`**. Check and perhaps edit this file. Some Linux distributions, especially 64 bit ones, do not use the standard directories for 32 bit executables. Under Ubuntu, you may have to use **`sudo`**.

```
$ ./InstallMe.Lin.sh
```

or

```
$ sudo ./InstallMe.Lin.sh
```

The cross compiler itself is in the `Compiler` directory and is usually called

```
x<cpu><type>Lin
```

where type is `Dev` or `Stamp`.

Now please read at least the installation and configuration chapter of `Docs/XC7man.pdf`. We do document our products.

Manuals

In order to reduce shipping costs and to provide you with up to date manuals we no longer supply printed manuals except on special request. Manuals are provided in PDF (and some in HTML formats) in the **Docs** directory. The **ProgRef** (or <cpu>) directory contains chip data sheets.

All the manuals and release notes are in the Docs directory. Please read them! The ones you need are in two groups, tools and target code.

- **XC7man.pdf** – the generic cross compiler manual for all versions
- **<cpu>man.pdf** – the CPU specifics
- **aide.pdf** – the manual for AIDE (Windows only)
- **CommonCode.pdf** – the generic target code manual
- **<cpu>Code.pdf** – the CPU specific target code manual

Running the compiler

Under Windows, the compiler is usually run from AIDE which serves as a configurable front end to the compiler and includes a (simple) editor and a terminal cum file server for the target Forth.

Under Linux and Mac OS X, the compiler can be run from the command line or using a shell script. The compiler parses the command line and treats everything after the compiler name as Forth source code.

AIDE (Windows)

The installer creates a pointer to AIDE. You can also make a short cut to AIDE.EXE to put on your desktop.

Run AIDE, and you will find that the Tools toolbar contains one or more entries for target configurations. Click on one of these and the compiler for that target will be run. Each target is set up using the IDE -> External Tools dialog. Note that to preserve any changes you must click the Apply button.

You can configure AIDE to use your favourite editor using the IDE -> Configure dialog.

The manual is `Docs\Aide.pdf`.

Linux and Mac OS X

The install script should have put the compiler and shared libraries in the right places, usually

```
/usr/bin
/usr/lib
```

You can then run the compiler by invoking its name and providing a command line, which is just treated as Forth source code, e.g.

```
Monet:LPC17xx $ xArmCortexDevLin include Olimex1766stk.ctl
```

When the compiler finishes, it stays alive so that you can disassemble and locate code while debugging your target.

Compiler

The general MPE Forth cross compiler manual is XC7man.pdf.

The CPU and target specific portions of the manual are in `Docs/<cpu>.pdf`.

Release notes are contained in the `RELEASE.XXX.TXT` files.

Macros

Both AIDE and the cross compiler support text macros for use in file names. AIDE expands its macros before passing them to the cross compiler. The point of the macros is to allow you to

a) define a directory path once
b) change it easily when moving a project from a desktop to a laptop.

Downloading to the target

The cross compiler produces output as a memory image file with no headers. These files can be used directly by most EPROM/Flash programmers and emulators. Tools are provided (see the cross compiler manual) for generating Intel Hex, Motorola S-Rectord files and some ELF formats..

The Forth Stamp implementations include downloaders for supported chips and the cable descriptions are provided with the hardware manuals.

Port Access

Some of the compilers (e.g. 8051 and AVR) support direct download through a PC printer port to chips with in-system program facilities. The compiler requires direct access to the printer port, which is not available by default under non-9x versions of Windows. The `COMPILER\XTRA` directory contains a port driver that permits direct access to the printer port. This must be installed before accessing the printer port under Windows NT/2000/XP. Installation instructions are in **`XC7man.pdf.`**

First Steps

Start by compiling one of the existing projects. Target code is in two folders

`x<CPU>/<cpu>`

and

`x<cpu>/Common`

Inside **`x<CPU>/<cpu>`** you will find either or both of the directories **`Configs`** and **`Hardware`**. The Configs folder holds control files which are equivalent to project files in other languages. A control file usually contains the hardware description of the project in terms of memory, stack sizes and son, plus a list of files to be included. Control files names usually have a **`.ctl`** extension. **`Hardware`** directories exist for CPU cores such as the ARM which have a huge number of variants and a wide range of peripherals specific to each supplier. Subdirectories within **`Hardware`** contain hardware specific code and control files.

We recommend that you start with an already supported evaluation board and its control file. Control files and compiler directives are covered in considerable detail in **`XC7man.pdf`**.

Then you can move on to generating your own control file. Start from the control file that is closest to your hardware configuration.

Additional tools

The TOOLS directory contains a selection of software tools that we have found useful for embedded systems development. More tools are added from time to time. Documentation for the tools will be found in their directories.

Source Code

If you want to know how the target works, the source code is available in the **<cpu>** and **Common** directories. If you change the MPE target code, please copy it to a separate file so that you will not lose your changes if you upgrade to a later version of the compiler and target code.

Source code for the non-VFX compilers is provided except for IRTC, Stamp and evaluation versions.

Source code for the VFX compilers is only released after signature of a Non Disclosure Agreement (NDA) and purchase of the full Developer edition of the compiler.

Technical support

Support is available from your supplier or directly from MPE. The contact details are at the top of this document. Telephone support is available during UK office hours.