MicroProcessor Engineering Limited

Forge - Forth Development Environment

Alpha Release Manual 15th September 1999

Forge - Forth Development Environment

Forge is an integrated development environment for the development and debugging of Windows based applications using MPE's ProForth VFX compiler suite.

Forge Integrates:

- On-line help
- Source File Editing
- Project/Build management
- Compiler tools
- Win32 Debugging (single-step machine level debugger)

Forge supports the edit/compile/execute cycle for both source files and complete projects (single or multiple targets).

You can choose the level of use of Forge to suit your own needs. Either use it to launch the debugger, use it to control source files and perform the build "by hand" in a standalone console, or use the integrated project management and compiler support to form a complete IDE.

First Use - Write a single file and Compile/Execute

This tutorial will show you how to use Forge to edit/compile/execute a single Forth source file.

1. Create a new blank source file.

Click on the New toolbar icon (or select `FILE->NEW` from the menu). A new blank source window will be opened.

2. Write some source code

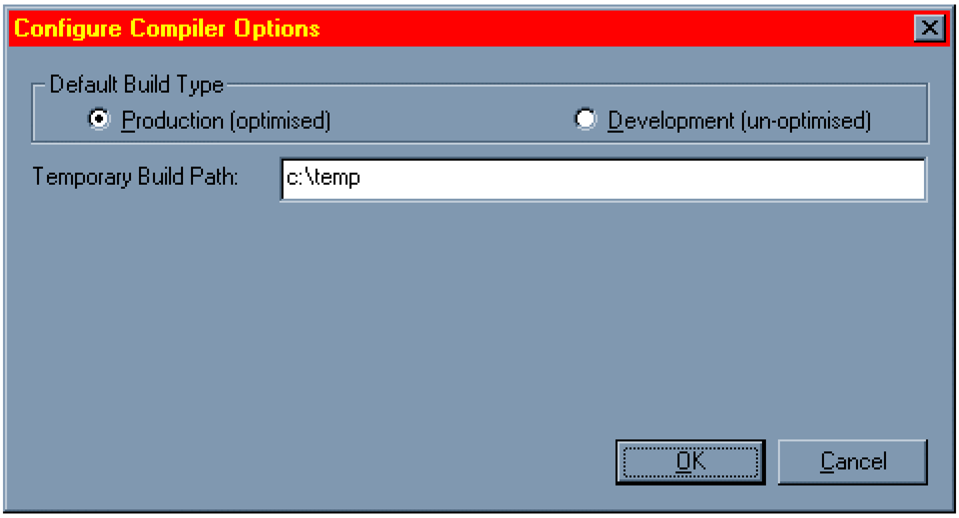
Enter the following definition into the new source window.

```
: .hello
  cr ." Hello world"
;
```

and save it with the Save toolbar icon (or `FILE->SAVE` as on the menu.)

3. Select compiler mode.

Click the compiler setup icon (or select `COMPILER->SETUP`) to raise the compiler options dialog box.



Configure Compiler Options

Default Build Type

● Production (optimised) ○ Development (un-optimised)

Temporary Build Path: c:\temp

This dialog box controls the VFX compiler when building single source files. You can select the temporary build path (where the code is placed after compilation) and the code generation type.

Release code will use the MPE optimiser; Development builds disable all optimisers to generate target code that can be easily recognised during debugging.

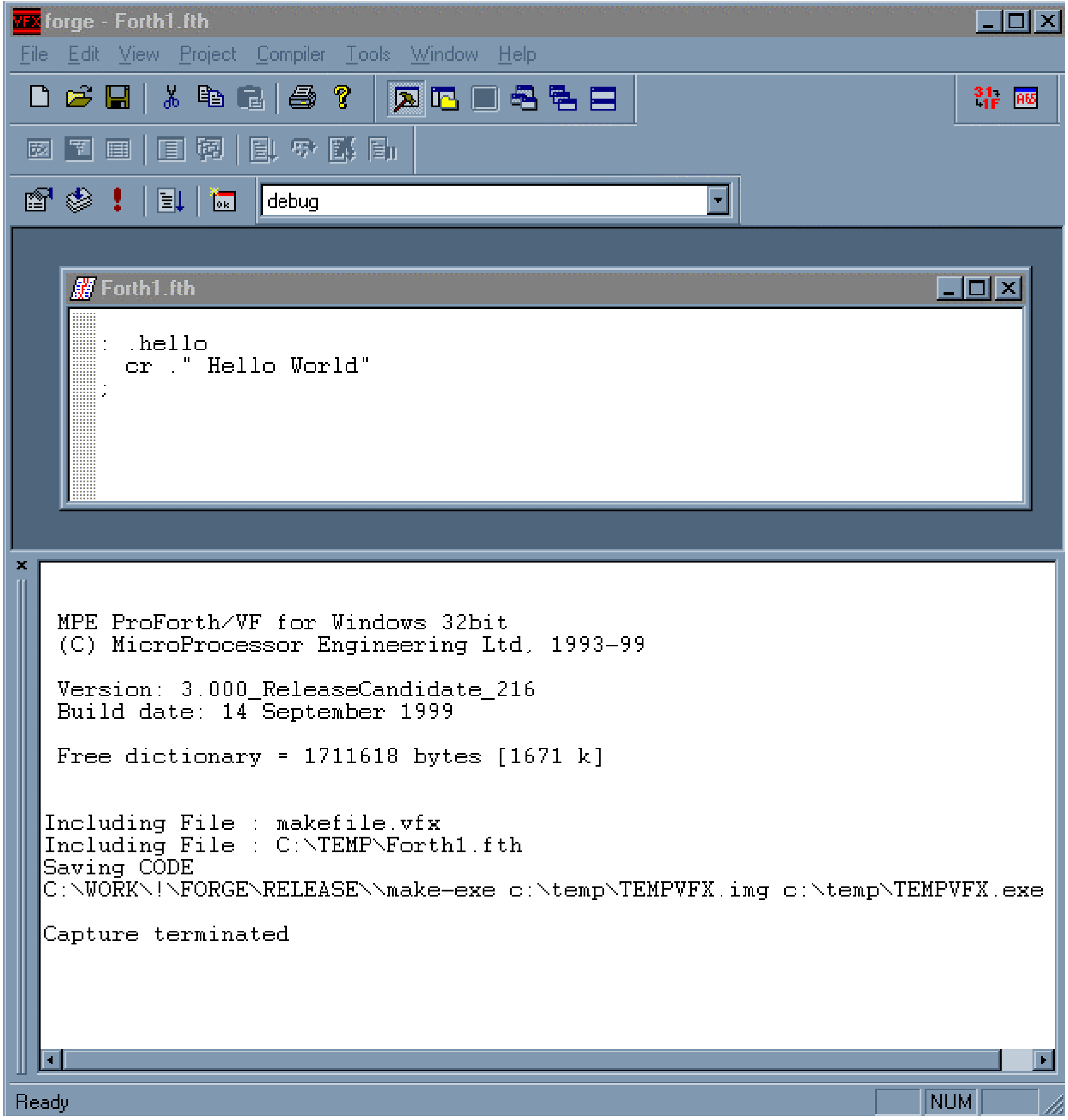
Ensure the build path is valid and select code generation type.

The options selected in the dialog will remain valid for any single file compilation until you change it. There is no need to repeat this step for every build unless you wish to change the options.

4. Compile your file.

After deciding the compilation type you can build the current source file with the compile icon.

The progress of the compiler can be seen in the compiler action window as seen below.



```
: .hello
  cr ." Hello World"
;
```

```
 MPE ProForth/VF for Windows 32bit
 (C) MicroProcessor Engineering Ltd, 1993-99

 Version: 3.000_ReleaseCandidate_216
 Build date: 14 September 1999

 Free dictionary = 1711618 bytes [1671 k]


Including File : makefile.vfx
Including File : C:\TEMP\Forth1.fth
Saving CODE
C:\WORK\!\FORGE\RELEASE\\make-exe c:\temp\TEMPVFX.img c:\temp\TEMPVFX.exe

Capture terminated
```

5. Run your code

! Once compilation has successfully completed you can execute your code by clicking on Execute Icon.

A ProForth console will popup. Enter

```
.HELLO <cr>
```

at the console to invoke your definition.

Building a Project

Forge also supports the concept of "projects". A project is a group of sources which are used to construct one or more related "targets".

This section will create a simple project which has two build targets. The first build target will be an optimised turnkey application, the second target will be a debug enabled version with an open interpreter.

1. Create a new workspace

A Forge project is stored as part of a "workspace". The workspace holds all related sources and build information necessary to generate various targets.

The workspace is distinctly split into two sections: -

1. Sources

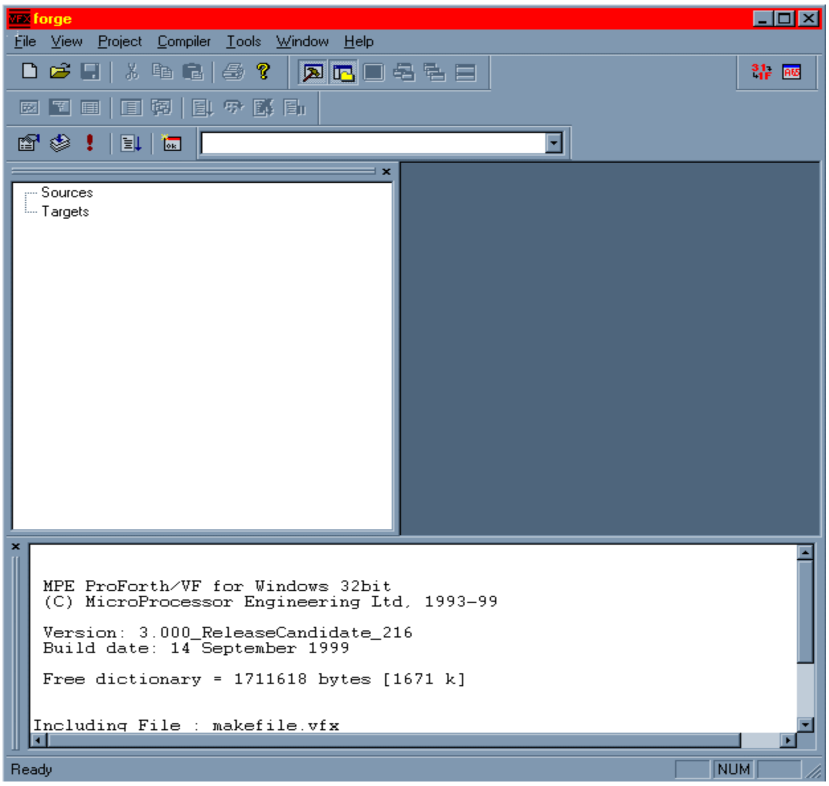
Any file, which is used as part of the project, must be referenced here.

2. Targets

Various build objects which are made from compiling one or more of the files in the SOURCES section.

A new workspace is generated from the menu option `FILE->NEW WORKSPACE`.

You should see the workspace panel appear as seen below, the two branches SOURCES and TARGETS are currently blank.



```
MPE ProForth/VF for Windows 32bit
(C) MicroProcessor Engineering Ltd, 1993-99

Version: 3.000_ReleaseCandidate_216
Build date: 14 September 1999

Free dictionary = 1711618 bytes [1671 k]

Including File : makefile.vfx
```

2. Create sources for use in project.

For this demonstration we will require two source files. The first file will be our test application, the second file will be the code necessary to make that application a turnkey system.

Source#1 - The application

Create a new file (as in the previous tutorial) and enter the text:

```
: MyTest        \ -- ; Popup a Messagebox HELLO WORLD
  HWND_DESKTOP Z" Hello World" Z" TestApp" MB_OK
  MessageBox drop
;

\ My "winmain" function

: MyStart       \ hInstance hPInstance szCommand nShow -- res
  2drop 2drop
  MyTest
  0
;
```

And save this file as "myapp.fth" on your harddisk.

Source#2 - The turnkey generation

Create another source file and enter the text:

```
\ Make application turnkey
assign MyStart to-do EntryPoint
```

And save this file as "mkturn.fth" on your disk.

3. Add files to project "available sources" list

In order to use the newly generated sources as part of the project it is necessary to add them to the workspace.

This can be done in one of two ways. You can either select the menu option `PROJECT->ADD FILES`, or you can right click on the "sources" branch in the project view window and select `ADD TO PROJECT`.

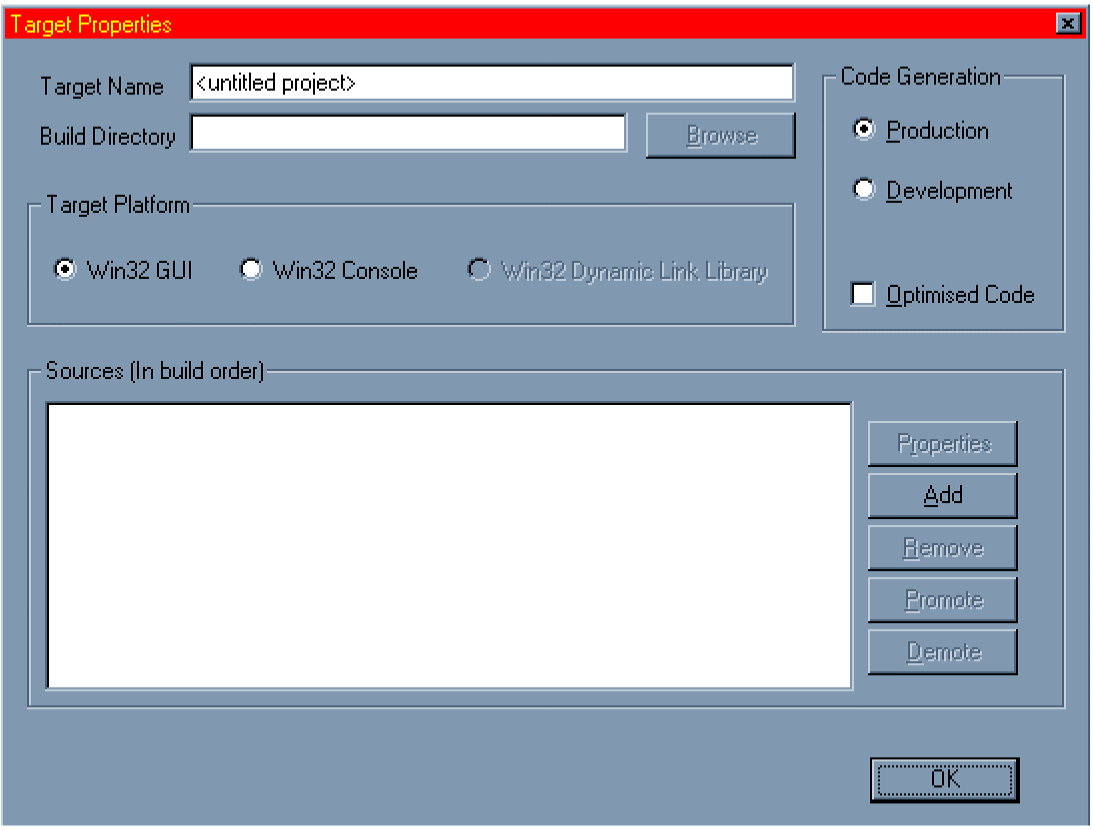
Select the file to add in the opened file selector and repeat for the second source.

By expanding the "sources" branch in the project window you will see the two files have been added to the available list. You can bring any source into the editor by double clicking its name.

4. Add build rules.

We are now going to add two build rules for a DEBUG and a RELEASE version of the project.

Select either `PROJECT->ADD NEW TARGET` or right click the "targets" section in the project window to invoke the "add new target" dialog as seen below.



Target Properties

Target Name: <untitled project>

Build Directory:

Target Platform: ● Win32 GUI ○ Win32 Console ○ Win32 Dynamic Link Library

Code Generation: ● Production ○ Development ☐ Optimised Code

Sources (In build order)

This dialog box is used to create or edit a target in the current workspace. It is divided into four sections.

1. Files/Directories

The TARGET NAME field is the executable name (without any file extension)

The BUILD DIRECTORY field is the directory from which the build will be executed and is the destination for the generated code.

2. Platform Selection

Here you can select between GUI CONSOLE or DLL generation.

3. Code Generation

Here you control the code generation strategy employed by the compiler when you build this target.

PRODUCTION code will generate code suitable for release to the general public. Among other things it ensure that Windows Exceptions are handled by the CATCH..THROW mechanism as described in the VFX Manual.

DEVELOPMENT code is just that. Code which is suitable for the developer. The major difference from PRODUCTION code is that the exception mechanism is exposed for use with the Win32 Debugger built into Forge (more later). Any target which you wish to debug should have this mode set.

OPTIMISED CODE, when checked will enable the VFX optimiser during compilation. Normally this would be checked for PRODUCTION builds and open for DEVELOPMENT builds. When the optimiser is disabled the actual machine code generated by the compiler will have an easily recognizable one to one correlation between Forth words in the source and machine code CALLS in the binary. With the optimiser enabled the binary loses this easy mapping and becomes much harder to debug.

4. The build sources

The bulk of the dialog box holds a list (currently empty) of the sources used (from the workspace "source" pool) to generate this target. They are listed in BUILD order (Ie the order in which they are compiled.)

The five buttons on the right are used to control the source listing.

PROPERTIES
Get information on the currently selected source file.

ADD
Add a new source from the workspace pool to the build.

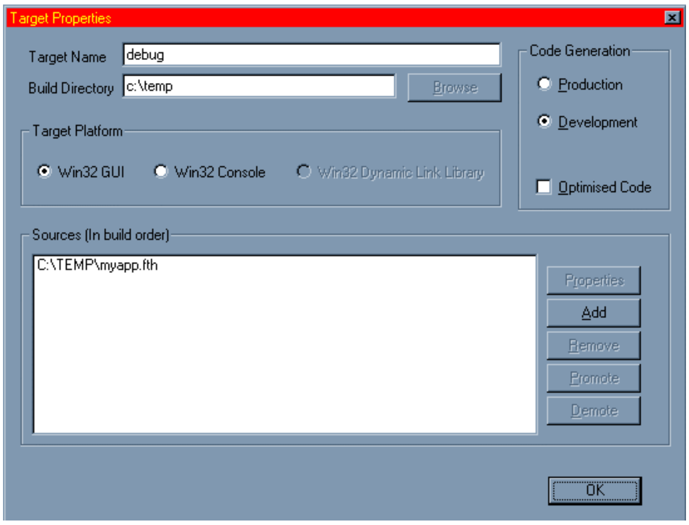
REMOVE
Remove the selected source from the build. This does **not** remove the source from the project active list or the harddisk.

PROMOTE
Move the selected source up one position earlier in the build order.

DEMOTE
Move the selected source one down in the build order (build later).

Now fill in the dialog box such that it looks like the image below.

This is the generation of the DEBUG target.



Target Properties

Target Name: debug

Build Directory: c:\temp

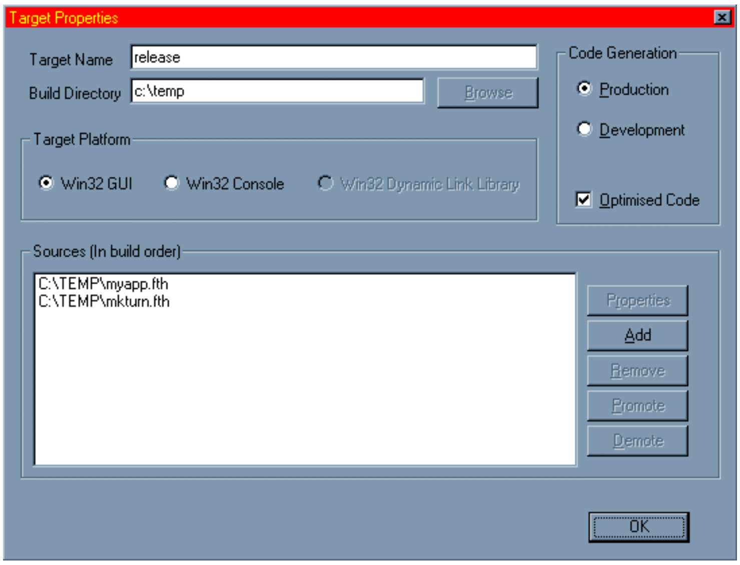
Target Platform: ● Win32 GUI ○ Win32 Console ○ Win32 Dynamic Link Library

Code Generation: ○ Production ● Development

☐ Optimised Code

Sources (In build order):
C:\TEMP\myapp.fth

And add a new target and fill in like so:



Target Name: release

Build Directory: c:\temp

Target Platform: ● Win32 GUI ○ Win32 Console ○ Win32 Dynamic Link Library

Code Generation: ● Production ○ Development

☑ Optimised Code

Sources (In build order)

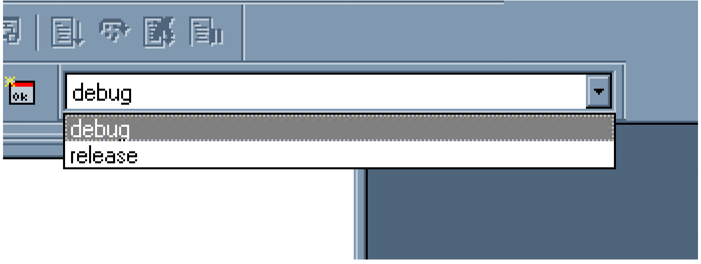
```
C:\TEMP\myapp.fth
C:\TEMP\mkturn.fth
```

which will generate the RELEASE target.

5. Compiling a Target

When a valid project is active the compile icon seen in the previous tutorial is now used to compile the active target rather than the active source file.

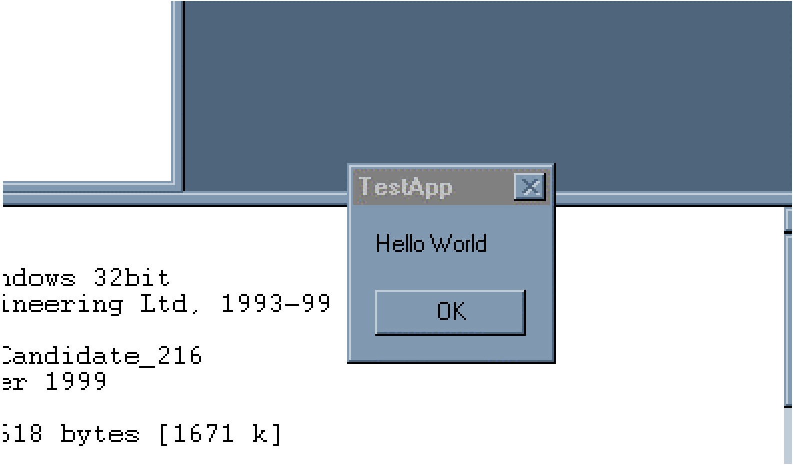
Which target is currently active can be selected by the drop down box in the toolbar as shown below:



Select the release version.

6. Execute Target

The Execute Icon is used to execute the currently selected target when a workspace is open. If you click on it now, the release version of the code we built in the previous step will execute. You should get a HELLO WORLD dialog box pop up as below:



Hello World

```
ndows 32bit
ineering Ltd, 1993-99

Candidate_216
er 1999

518 bytes [1671 k]
```

The debug version can also be compiled and run in the same manner.

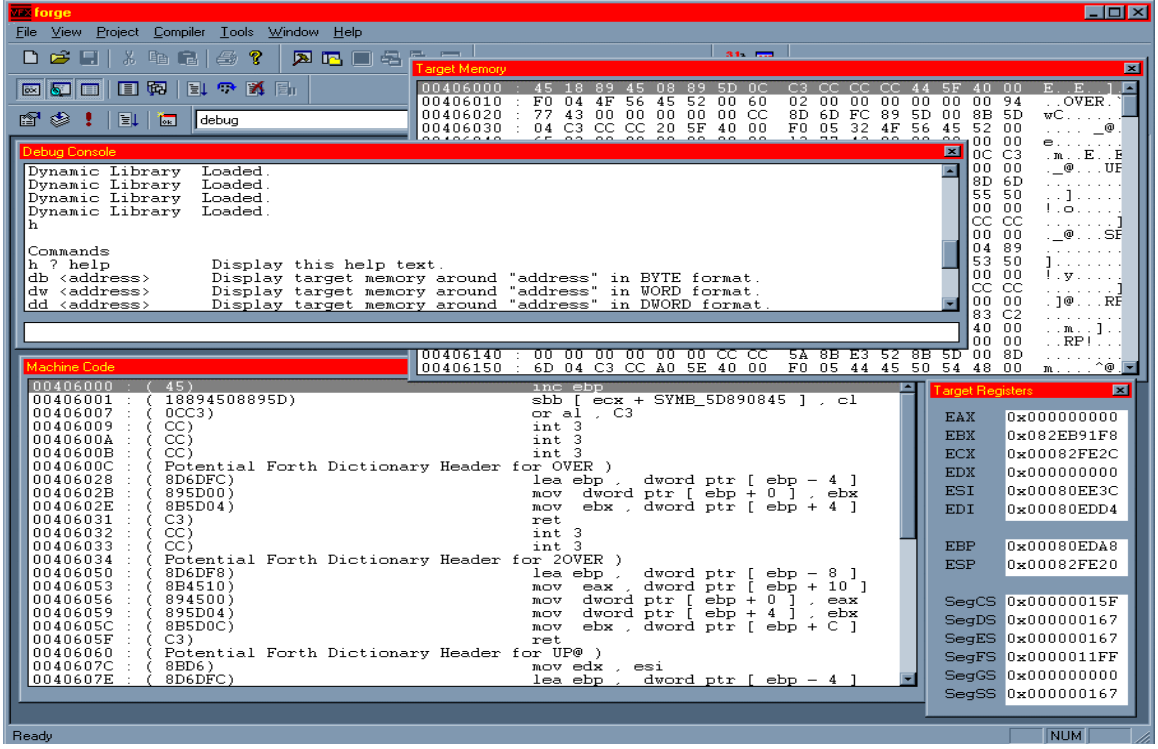
This time instead of the application running you should get a Forth console window popup which contains your code. (Remember we didn't add the MKTURN.FTH file to the debug build so it does not auto-run the application.)

Be aware that windows exceptions under the debug build are *not* trapped by **CATCH** and **THROW**. For this you need to use the Forge Debugger.

Debugging with FORGE

Forge contains a Win32 debugger. The debug icon can be used to run the currently active target under the control of the debugger. The system can handle **any** Win32 Code and specifically handles VFX constructs (such as dictionary headers and label name resolution.)

When active the debugger can display a number of information screens.



```
00406000 : 45 18 89 45 08 89 5D 0C  C3 CC CC CC 44 5F 40 00  E..E..]..
00406010 : F0 04 4F 56 45 52 00 60  02 00 00 00 00 00 00 94  ..OVER.`
00406020 : 77 43 00 00 00 00 00 CC  8D 6D FC 89 5D 00 8B 5D  wC......
00406030 : 04 C3 CC CC 20 5F 40 00  F0 05 32 4F 56 45 52 00  ...._@.
00406140 : 00 00 00 00 00 00 CC CC  5A 8B E3 52 8B 5D 00 8D  ........
00406150 : 6D 04 C3 CC A0 5E 40 00  F0 05 44 45 50 54 48 00  m....^@.
```

```
Dynamic Library  Loaded.
Dynamic Library  Loaded.
Dynamic Library  Loaded.
Dynamic Library  Loaded.
h

Commands
h ? help          Display this help text.
db <address>      Display target memory around "address" in BYTE format.
dw <address>      Display target memory around "address" in WORD format.
dd <address>      Display target memory around "address" in DWORD format.
```

```
00406000 : ( 45)                                   inc ebp
00406001 : ( 18894508895D)                         sbb [ ecx + SYMB_5D890845 ] , cl
00406007 : ( 0CC3)                                 or al , C3
00406009 : ( CC)                                   int 3
0040600A : ( CC)                                   int 3
0040600B : ( CC)                                   int 3
0040600C : ( Potential Forth Dictionary Header for OVER )
00406028 : ( 8D6DFC)                               lea ebp ,  dword ptr [ ebp - 4 ]
0040602B : ( 895D00)                               mov  dword ptr [ ebp + 0 ] , ebx
0040602E : ( 8B5D04)                               mov  ebx , dword ptr [ ebp + 4 ]
00406031 : ( C3)                                   ret
00406032 : ( CC)                                   int 3
00406033 : ( CC)                                   int 3
00406034 : ( Potential Forth Dictionary Header for 2OVER )
00406050 : ( 8D6DF8)                               lea ebp ,  dword ptr [ ebp - 8 ]
00406053 : ( 8B4510)                               mov  eax , dword ptr [ ebp + 10 ]
00406056 : ( 894500)                               mov  dword ptr [ ebp + 0 ] , eax
00406059 : ( 895D04)                               mov  dword ptr [ ebp + 4 ] , ebx
0040605C : ( 8B5D0C)                               mov  ebx , dword ptr [ ebp + C ]
0040605F : ( C3)                                   ret
00406060 : ( Potential Forth Dictionary Header for UP@ )
0040607C : ( 8BD6)                                 mov edx , esi
0040607E : ( 8D6DFC)                               lea ebp ,  dword ptr [ ebp - 4 ]
```

```
EAX     0x000000000
EBX     0x082EB91F8
ECX     0x00082FE2C
EDX     0x000000000
ESI     0x00080EE3C
EDI     0x00080EDD4

EBP     0x00080EDA8
ESP     0x00082FE20

SegCS   0x00000015F
SegDS   0x000000167
SegES   0x000000167
SegFS   0x0000011FF
SegGS   0x000000000
SegSS   0x000000167
```

The debugger control Icons are:

Show/Hide the register display/edit window. You cannot change register contents whilst the target is running.

Show/Hide the code window. This window displays a "Forth aware" disassembly.

Show/Hide the target memory window. This window displays a memory dump view under control of the debug console.

Resume Target Execution. When available this icon will cause the target to continue execution. If this option is grayed out then the target is already running (you may need to ALT+TAB to see it)

Single Step. When the target is halted (via user action or exception) this icon will allow you to step one machine instruction on. The code view window if active will automatically update as will the register display.

Terminate Target. If you wish to kill the target application you can use this option. Please note that Windows can get upset when you KILL processes. If possible you should close the target application normally and use this option in dire emergencies.

Suspend Target. This button will halt the running target and activate the debugger.

The Debug Console

The debug console displays various status messages as well as providing a command prompt (at the bottom of the window) for entering user commands.

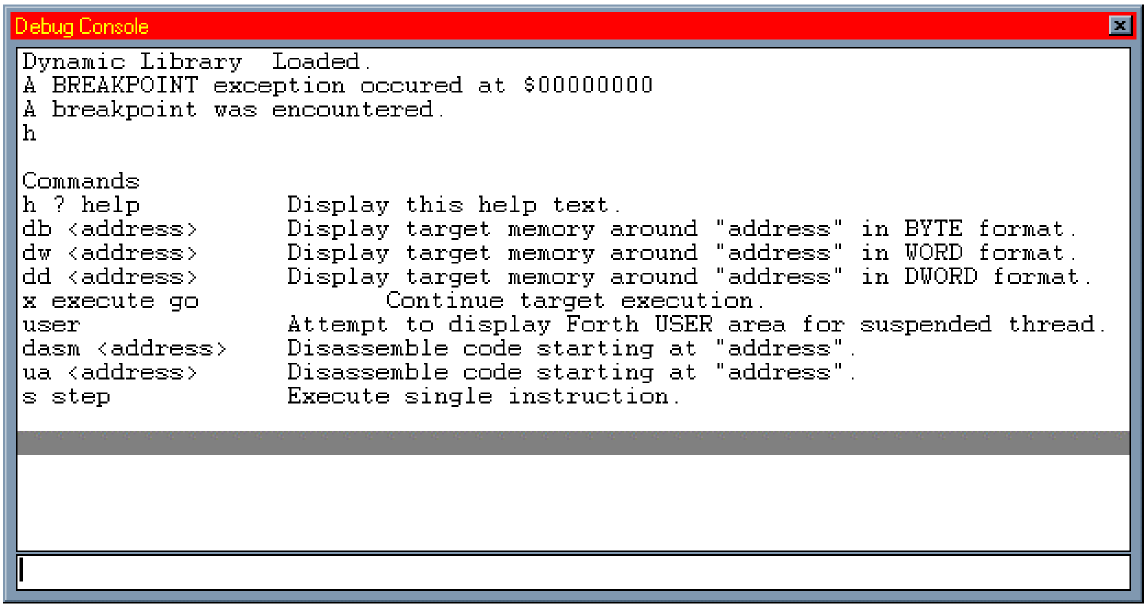
The command `HELP` will display available options. Please note that the step and execute instructions are only meaningful for a suspended target. If the target is already running they will have no effect.

Some commands (such as `DD, DB, UA` etc) require an address parameter. This can either be in decimal or in "C" style hex. IE:

```
DD      0x0400000 <CR>
```

Will perform a memory dump in `DWORD` format starting at the target address `0x0400000`.

You can use the memory display and disassembly commands on a running target at any time.



```
Dynamic Library  Loaded.
A BREAKPOINT exception occured at $00000000
A breakpoint was encountered.
h

Commands
h ? help          Display this help text.
db <address>      Display target memory around "address" in BYTE format.
dw <address>      Display target memory around "address" in WORD format.
dd <address>      Display target memory around "address" in DWORD format.
x execute go             Continue target execution.
user              Attempt to display Forth USER area for suspended thread.
dasm <address>    Disassemble code starting at "address".
ua <address>      Disassemble code starting at "address".
s step            Execute single instruction.
```

Other Icons and Options

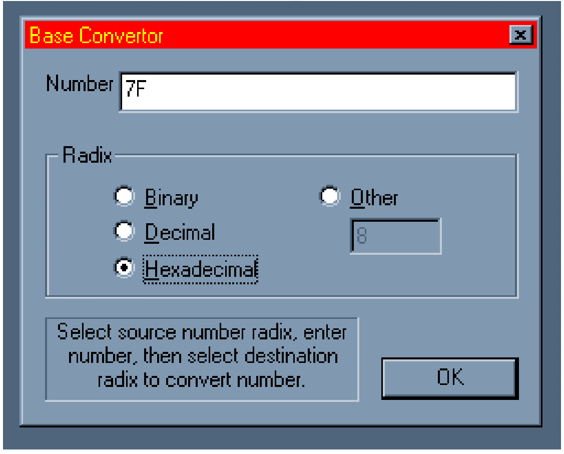
Forge is littered with various other options and switches, here are the ones which need explaining.

Compiler Setup

This dialog box configures the use of the Proforth compiler for single file builds (as seen in the first tutorial). You select the default code generation type and build path.

Number Convertor

This simple tool allows for the conversion of integer numbers from one base to another.



Number 7F

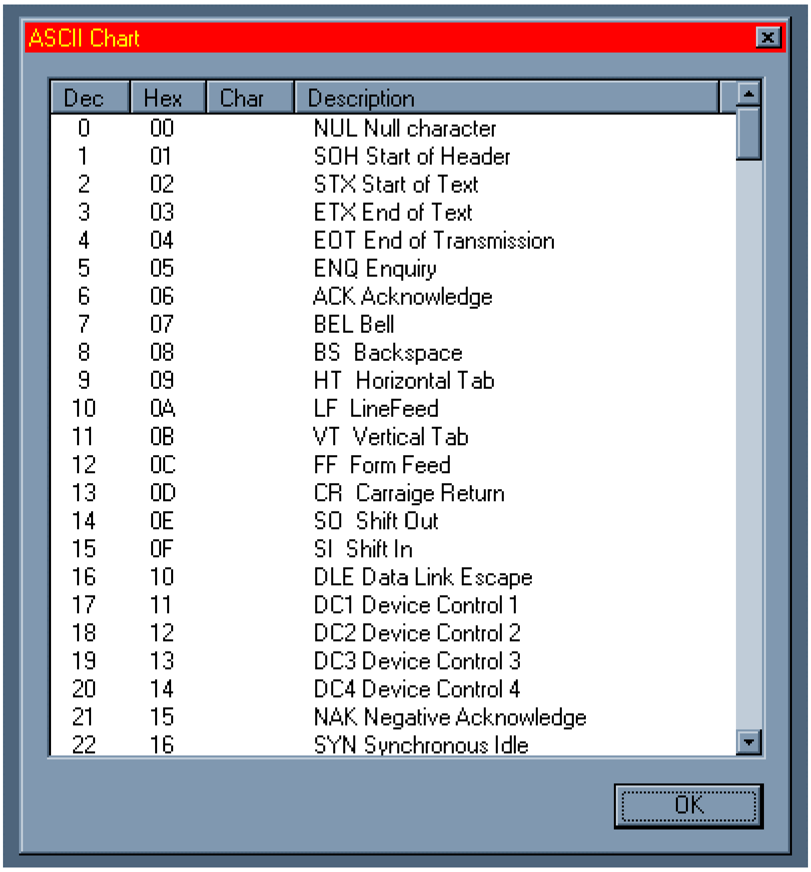
Radix: ● Hexadecimal

Other: 8

Select source number radix, enter number, then select destination radix to convert number.

ASCII Chart

Popups up a simple ASCII chart for reference.



Dec	Hex	Char	Description
0	00		NUL Null character
1	01		SOH Start of Header
2	02		STX Start of Text
3	03		ETX End of Text
4	04		EOT End of Transmission
5	05		ENQ Enquiry
6	06		ACK Acknowledge
7	07		BEL Bell
8	08		BS Backspace
9	09		HT Horizontal Tab
10	0A		LF LineFeed
11	0B		VT Vertical Tab
12	0C		FF Form Feed
13	0D		CR Carraige Return
14	0E		SO Shift Out
15	0F		SI Shift In
16	10		DLE Data Link Escape
17	11		DC1 Device Control 1
18	12		DC2 Device Control 2
19	13		DC3 Device Control 3
20	14		DC4 Device Control 4
21	15		NAK Negative Acknowledge
22	16		SYN Synchronous Idle

Spawn New Console

This icon will run a standalone ProForthVFX console window. It is not tied to the Forge environment and can be used as normal.

The Future

This Forge release is Alpha software. It is an indication of the future of the ProForth suite and is still very much in development.

Future improvements will include (in no particular order):

1. General user interface tidy up (window management etc.)
2. Improved debug abilities.
3. Addition of a Dialog resource editor.
4. Bitmap/Icon and Menu editors
5. RCS style version control.
6. Replacement of the editor with Scribe, a programmable editor with Syntax coloring, folding views etc.
7. User tool support.
8. ActiveX “sandbox” for testing.
9. Incremental compilation.
10. Support for the DOCGEN system (see the VFX Manual)
11. Locate/XREF support.

Any feature requests or (polite!) comments will be gratefully received at

forge@mpeltd.demon.co.uk

Go on! Have a play and see what you think! ;-)