

Dynamic C 8 Highlights

Overview

The primary improvement in Dynamic C 8 over previous versions is in the user interface. Most customer complaints about UI features that were lacking or hard to use are addressed in this release. The most notable of the UI improvements is the new editor, but many improvements have been made in debug features, ease of use and on-line documentation. The most notable non-UI improvements are the introduction of compiler listing files, a facility to compress imported files at compile-time and decompress them at run-time, and improvements to generate more compact code.

Note: Several improvements to the libraries and compiler have been, but this document mainly deals with user interface changes.

Summary of Improvements

- Full feature programming editor
- Easier to use, more configurable interface
- Flyover over watch expressions
- On-line I/O register reference
- More shortcut keys

Full feature editor

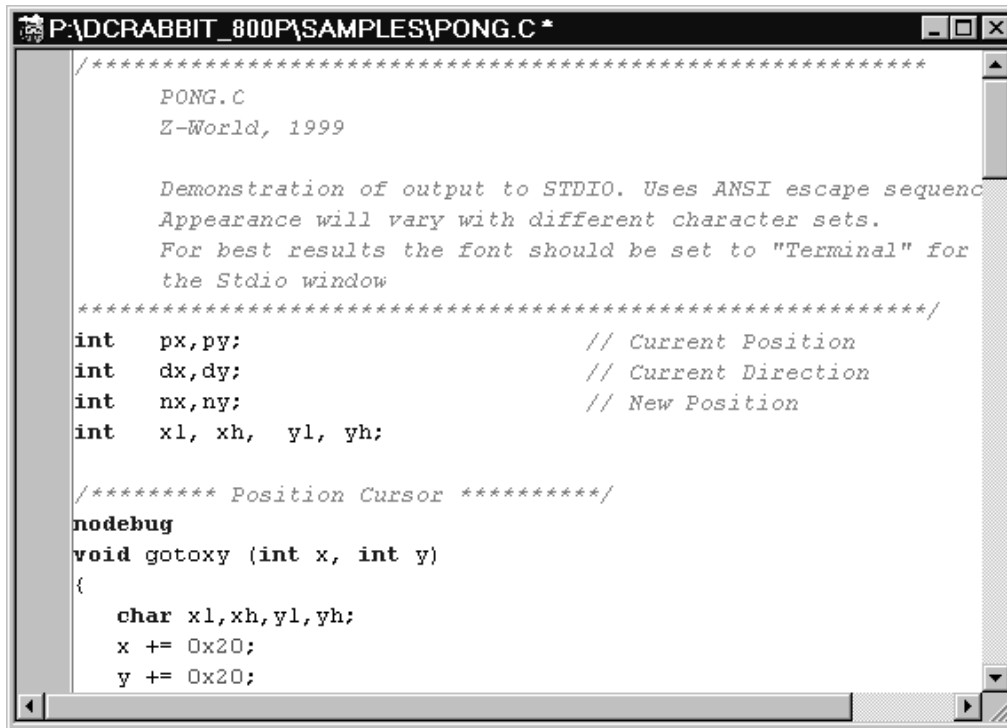
Dynamic C until now has had a simple, NotePad-type editor. Dynamic C 8 introduces a large set of new editor features to facilitate program writing. These include:

- Syntax highlighting
- Configurable code templates
- Column mode
- Bookmarks
- Parenthesis and curly brace matching
- Choice of several editor shortcut key schemes
- Multiple file search with regular expression handling (grep)

Syntax highlighting

There have been many requests for syntax highlighting. This feature allows different elements of source code (i.e. comments, variable names, constants etc.) to appear with different text attributes. The text attributes are completely configurable by the user. The default scheme is shown in figure Figure 1.

Figure 1. Editor window with syntax highlighting

A screenshot of a code editor window titled "P:\DCRABBIT_800P\SAMPLES\PONG.C *". The code is written in C and features syntax highlighting: keywords like 'int', 'void', and 'char' are in bold; comments are in italics; and string literals are in quotes. The code includes a header section with a multi-line comment, variable declarations for position and direction, and a function definition 'void gotoxy (int x, int y)'.

```
/* *****  
PONG.C  
Z-World, 1999  
  
Demonstration of output to STDIO. Uses ANSI escape sequence  
Appearance will vary with different character sets.  
For best results the font should be set to "Terminal" for  
the Stdio window  
***** */  
  
int  px,py;           // Current Position  
int  dx,dy;           // Current Direction  
int  nx,ny;           // New Position  
int  xl, xh,  yl, yh;  
  
/***** Position Cursor *****/  
nodebug  
void gotoxy (int x, int y)  
{  
    char xl,xh,yl,yh;  
    x += 0x20;  
    y += 0x20;
```

Templates

Code templates are a convenient way of inserting frequently used patterns of code or comments into source code. The cursor can be put on the place in the source code where the code snippet is to go, the right mouse button clicked and **Insert Code Template** chosen from the pop-up menu that appears as shown in Figure 2. A list of available templates appears, the user clicks the desired template, and the code snippet is inserted. A useful set of default templates will be included with Dynamic C, and users may add, edit and delete templates as they wish from the Options Menu. The template manager interface dilaog box is shown in Figure 3.

Figure 2. Inserting template text into a source file

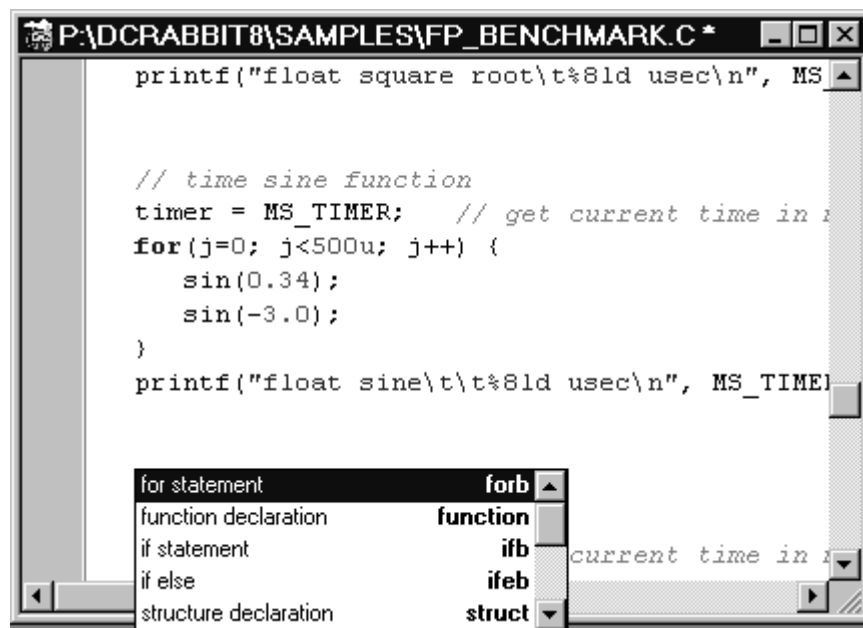
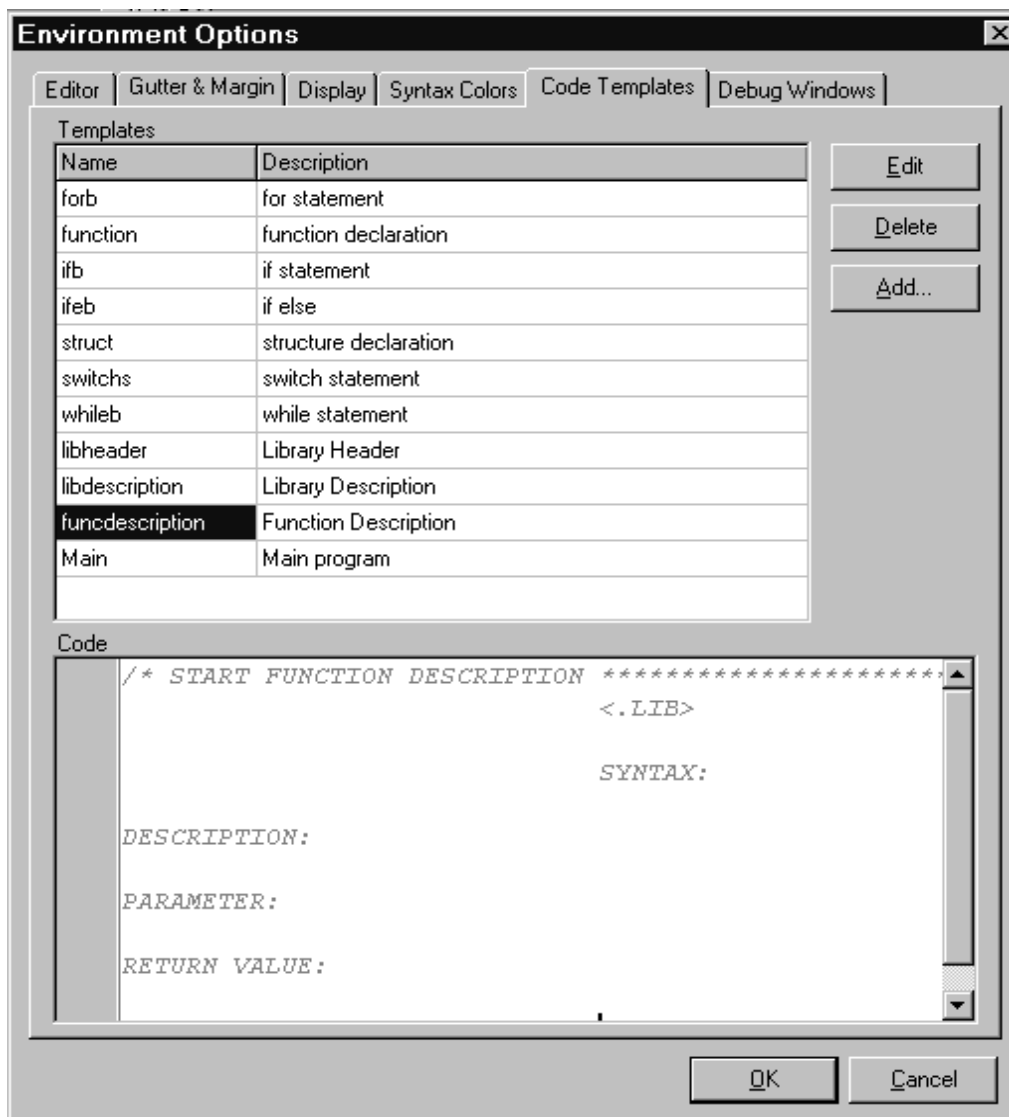


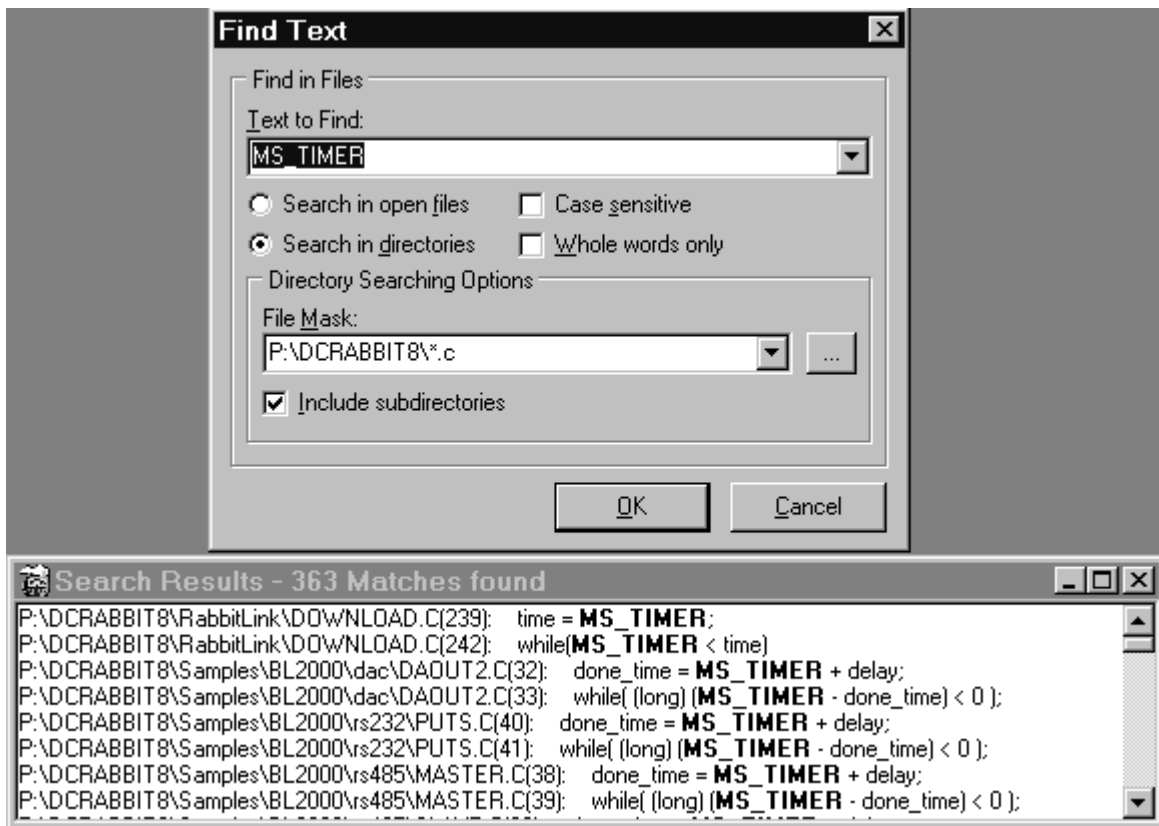
Figure 3. Interface for creating and modifying templates



Grep utility

Grep is a multiple file string search utility familiar to Unix and Linux users that is built into Dynamic C 8. Regular expressions (advanced, flexible wildcard searches) can be used. A window with a list of search results appears when the search is done, and clicking on a line in the results window opens the file referenced on the line and highlights the line the string is found on. Figure 4 shows examples of the grep dialog and search results windows.

Figure 4. grep interface and search results window



Column mode

Column mode allows editor text to be selected, copied, cut and pasted by column rather than row. Also, short cut keys allow selected text (whether selected by column or row) to be shifted right or left which makes it very easy to change indentation on code blocks.

Bookmarks

Bookmarks allow reference points to be set in a source code file that can be jumped to quickly. For example, a place 1000 lines into a file can be marked by typing Ctrl-Shift-0. To jump that place, the user type Ctrl-0.

Parenthesis and curly brace matching

To help find mismatched open/close curly braces and parenthesis, the user can place the cursor before the opening brace or parenthesis and type Ctrl-[. The cursor will jump to the matching end delimiter.

Several available keyboard shortcut schemes

Choose from:

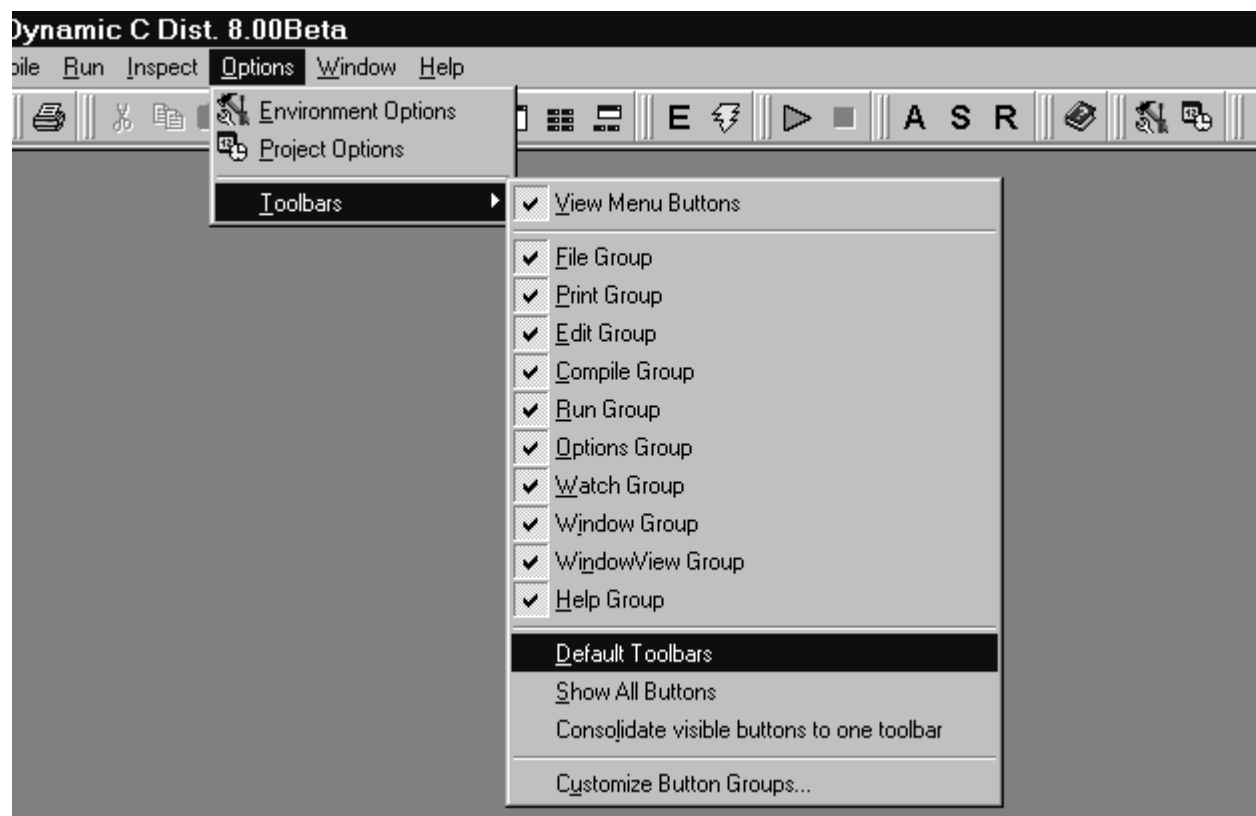
- Default
- Classic
- Brief

- Epsilon
- Visual Studio

Increased Configurability and Improved Options Interface

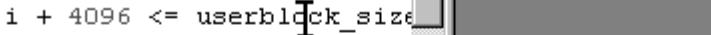
Many new options have been added for various features. The Options menu is better organized to distinguish between project level and environmental level options. The Environment and Project Options menu items will bring up easy to use tabbed dialog boxes. The buttons toolbar is now configurable to allow any and all menu commands to be assigned a toolbar button. The default button configuration is nearly identical to the that of the current version of Dynamic C, but a Run and Stop button have been added. Many of the buttons have changed appearance to give a more standard, modern Windows application appearance.

Figure 5. Options menu and button configuration interface



Watch expression interface improvements

Watch expressions can now be added or deleted by clicking the right mouse button in the watch window itself as shown in Figure 7. In addition, the values of variables can be examined simply by holding the cursor over the variable for a moment. A box showing the variable's name and value will pop up as shown in Figure 6.



The screenshot shows a debugger window with a code snippet and a variable watch window. The code snippet is:

```
long)i + 4096 <= userblock_size  
lock(i, buffer, 4096);
```

The variable watch window shows the variable `userblock_size` of type `long` with a value of `16252 (0x00003F7C)`.

The screenshot shows two windows from the Visual Studio IDE:

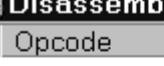
- Code Window:** Displays a C++ loop:

```
for (i = 0; i < userblock_size; i += 4096) {  
    if ((unsigned long)i + 4096 <= userblock_size)  
        writeUserBlock(i, buffer, 4096);  
}  
else {
```
- Watches Window:** Shows variables being monitored:

Name	Type	Value
buffer	char *	(0xB310) "\\x\F\x\F\x\F\x\F"
i	unsigned int	0 (0x0000)

An "Add Watch Expression" dialog box is open in the foreground, with "calib_size" entered in the "Watch Expression" field.

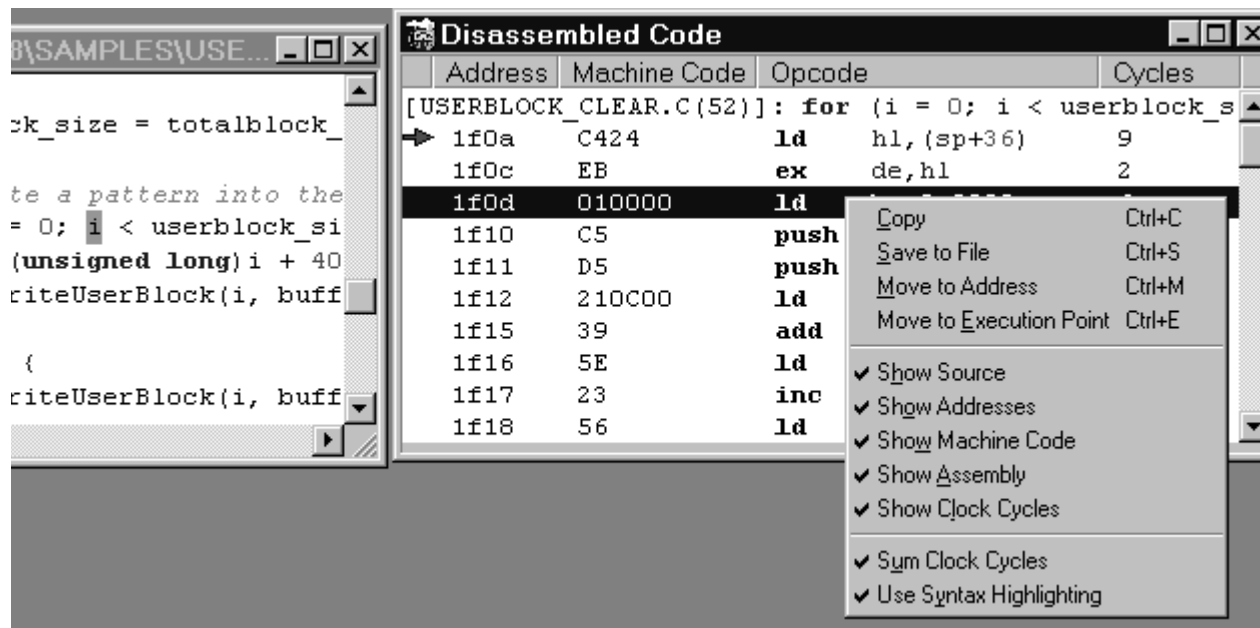
The assembly window has new functionality that includes the option to display C source code lines, and a **Run to Address** command. The window can be configured to show hex addresses, source code lines, machine code instructions, instruction mnemonics, and cycle times, or to show the minimal information of just mnemonics:



Disassembled Code

Opcode
call cls
rst 0x28
ld hl, 0x0001
ld (0xC302), hl
rst 0x28
ld hl, 0x0019
ld (0xC300), hl

Figure 9. Assembly window showing all information and right mouse pop-up menu

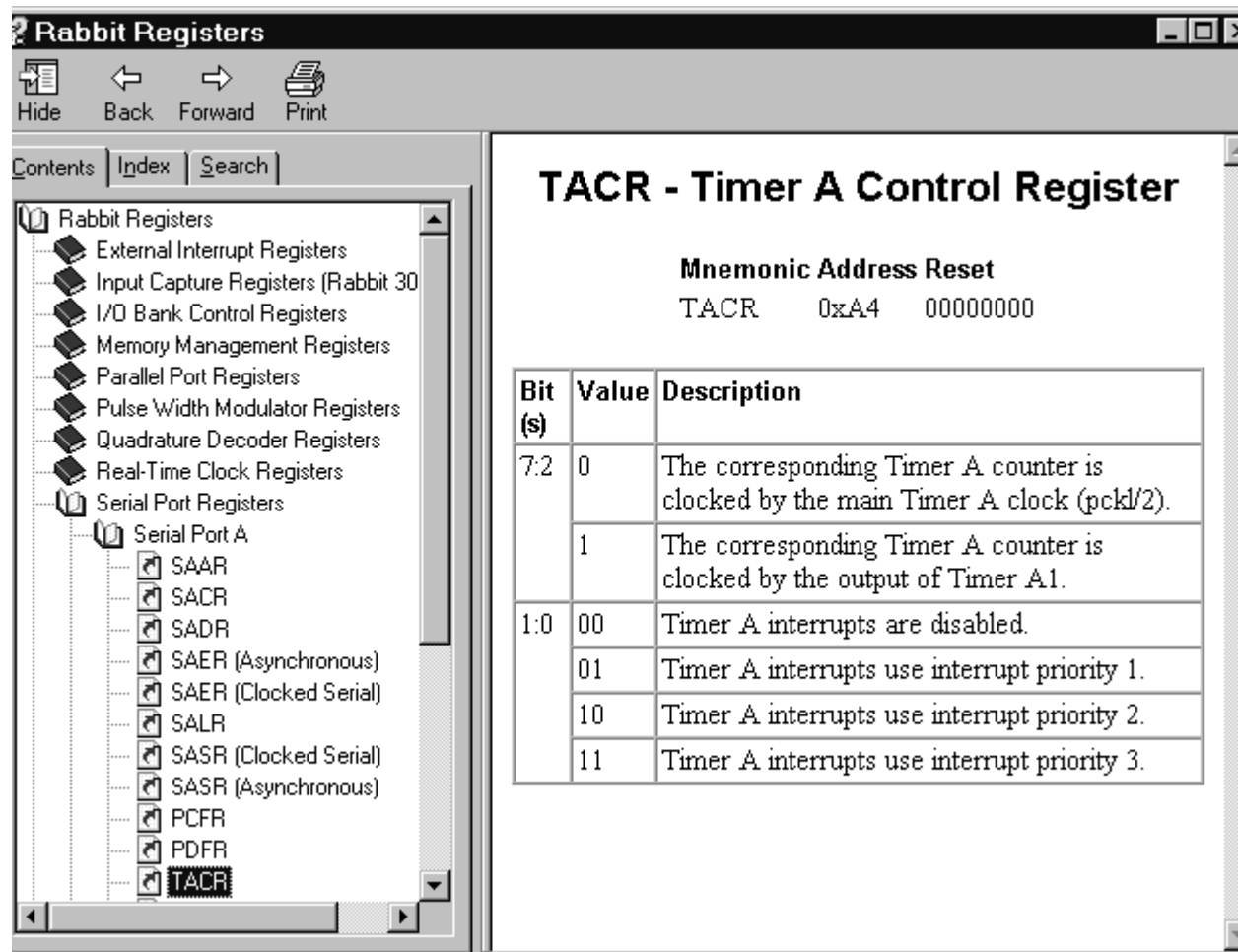


On-line Documentation Improvement

Rabbit internal I/O register reference

The Help menu has a new command called I/O Registers that brings descriptions of the I/O registers nicely organized by peripheral. Figure 10. shows an example this.

Figure 10. On-line I/O register help



Other interface improvements

Other debug windows have some improvements for ease of use also. The right mouse button has new uses in many debug windows. A variety of keyboard shortcuts too numerous to list here are documented in the on-line help.