UNIVERSITY OF THE PACIFIC

HOME Running Your Code>>

Getting Started with the Texas Instruments ez430

Working with the Workbench Software

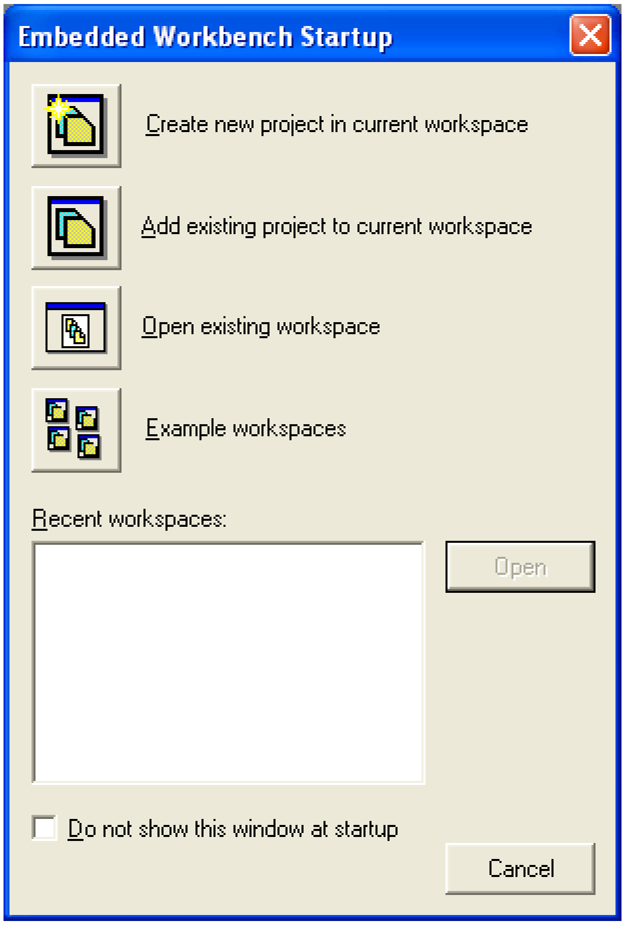
Step 1: Each program needs an associated project. The project includes your 'C' file as well as other files created by the system (a project file with the ".eww" extention, a project settings folder, a debug folder which contains compiled code outputted by Workbench etc).

If you save and close your project, opening it again is as simple as opening this folder, and doubleclicking on the Workbench project file, which has a ".eww" extention.

Using 'Start->My Computer', create a folder to store all your projects named "ez430". NOTE: You should create the folder in your U: drive. Create a folder in the "ez430" folder named "blink" for your first project. Each project should be saved in its own folder (directory).

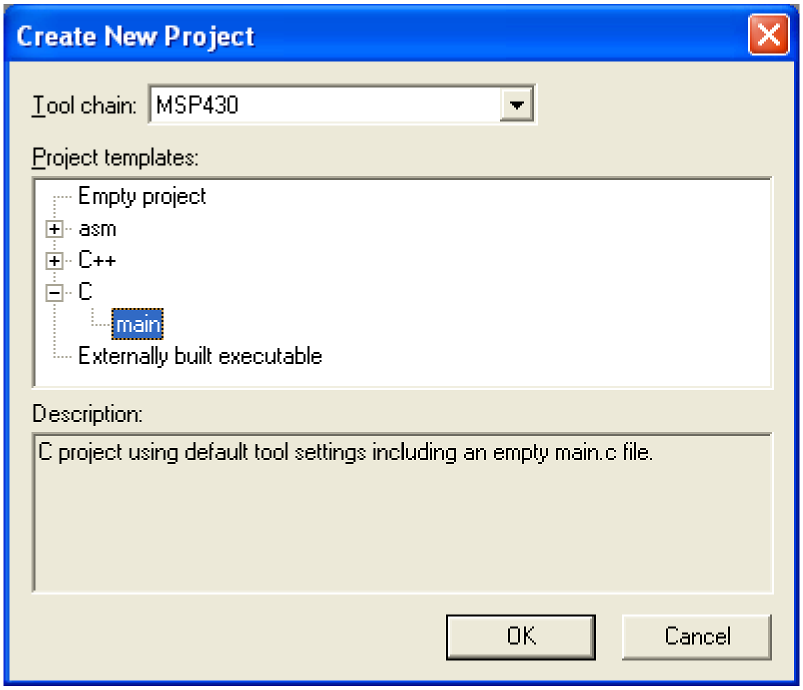
Step 2: KickStart provides an IDE (integrated development environment) where you can configure the hardware, write code, compile code, download and debug code. Start KickStart from the Start menu: 'Start-> Programs -> IAR Systems -> IAR Embedded Workbench KickStart for MSP430 V4 -> IAR Embedded Workbench'

A **Startup Screen** should open. If it does not open, choose: 'Help -> Startup Screen'


Embedded Workbench Startup
Create new project in current workspace
Add existing project to current workspace
Open existing workspace
Example workspaces
Recent workspaces:
Open
☐ Do not show this window at startup
Cancel

Step 3: Select "Create new project in current workspace" to begin.

A **Create New Project Screen** will open.



C project using default tool settings including an empty main.c file.

Step 4: Select the language in which we shall write our code. Click the '+' next to the 'C' and then the 'main' option. Click 'OK'. A 'Save As' window will open. Select the project folder "blink", you created earlier. Enter the File name "blink_project".

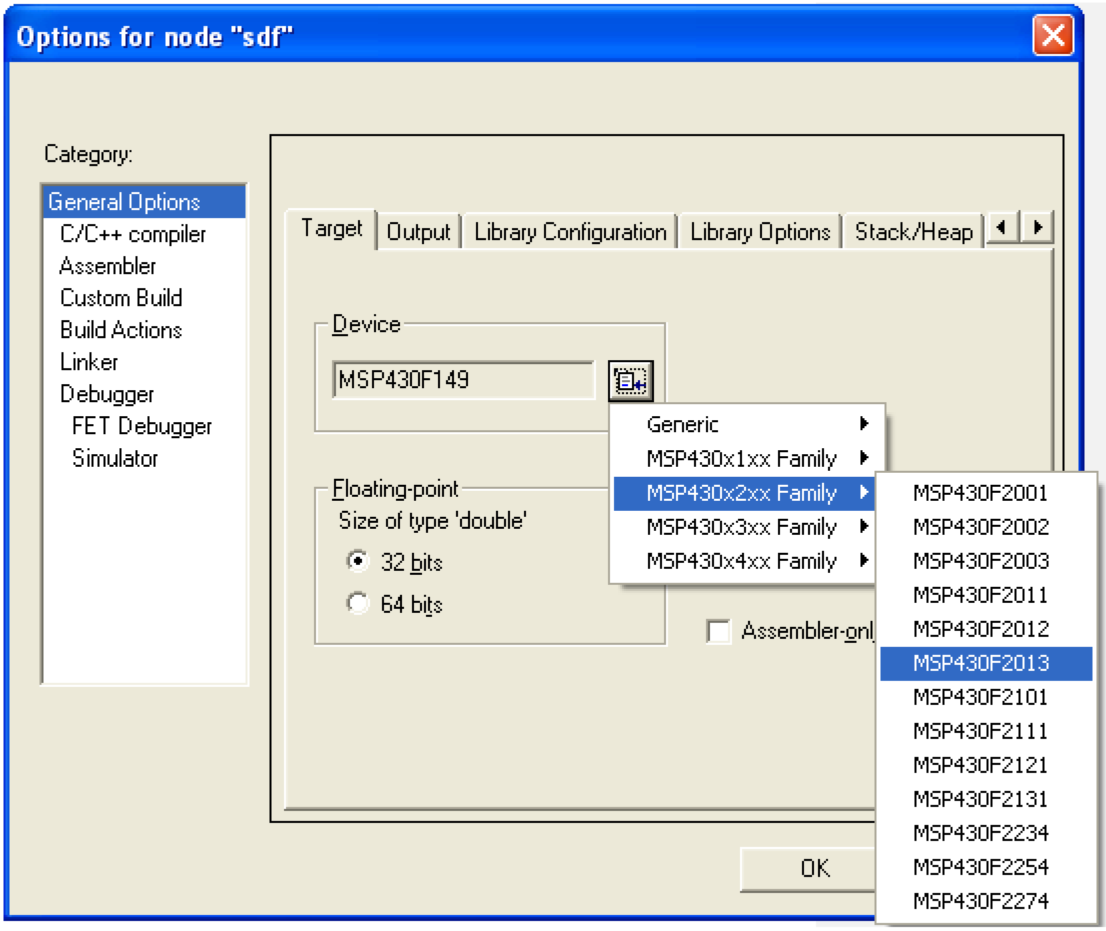
NOTE: You may get a system message letting you know that the project file is an old format, asking if you would like to convert it for use with this version. Select 'Yes'.

Configure Project

Once you have created your project folder, the IDE window is displayed with the **Files** section on the left column and the source code template **main.c** displayed in the main window. You now need to configure the project.

Step 5: Under the Files pane, select (highlight) the Project (top entry). Select 'File->Save Workspace'. Make sure you are saving in the project folder you created. Enter the project name "blink_workspace".

Step 6: Select **Project->Options** to get the following screen and make the indicated selections:



Options for node "sdf"

Category:

General Options

Device: MSP430F149

Floating-point

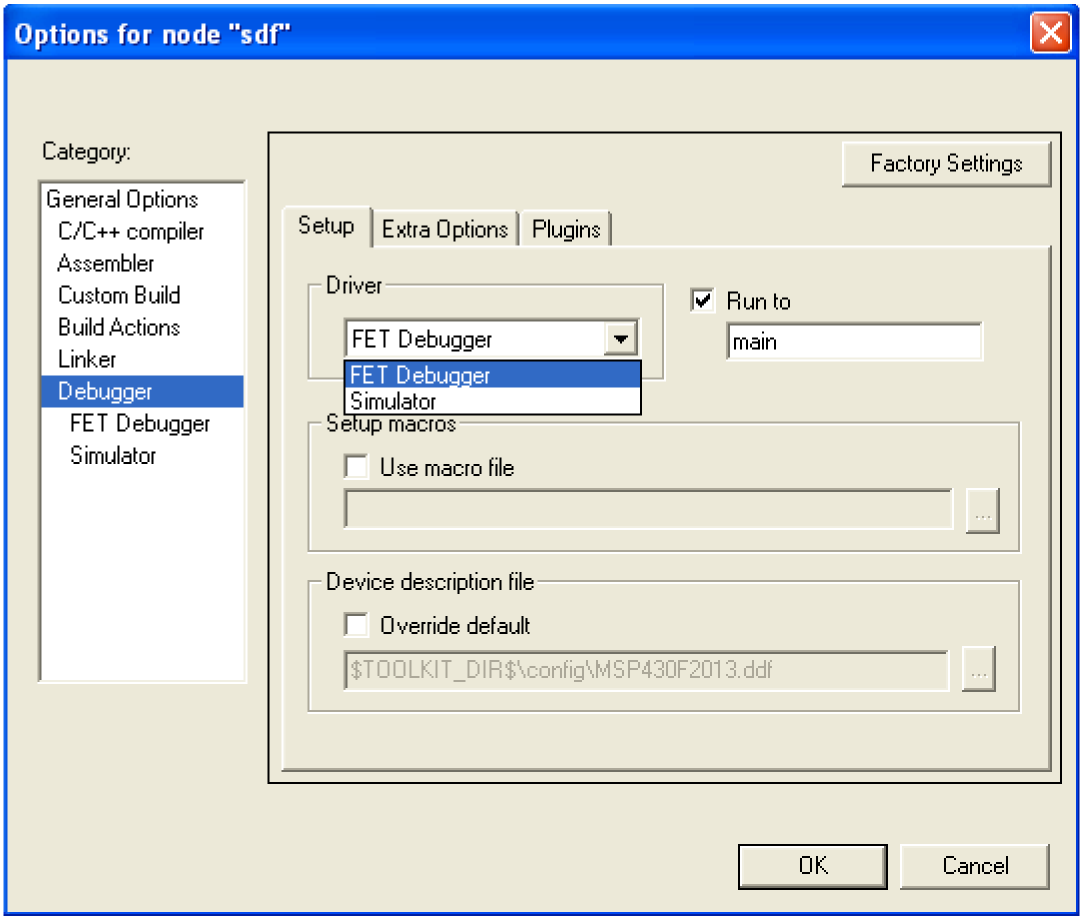
Size of type 'double'

● 32 bits

○ 64 bits

□ Assembler-onl

Step 7: Select Debugger in the Category section (left column of the Options window) and make the following selections.



Options for node "sdf"

Category:

- General Options
- C/C++ compiler
- Assembler
- Custom Build
- Build Actions
- Linker
- Debugger
- FET Debugger
- Simulator

Factory Settings

Setup | Extra Options | Plugins

Driver: FET Debugger (FET Debugger, Simulator)

☑ Run to: main

Setup macros

□ Use macro file

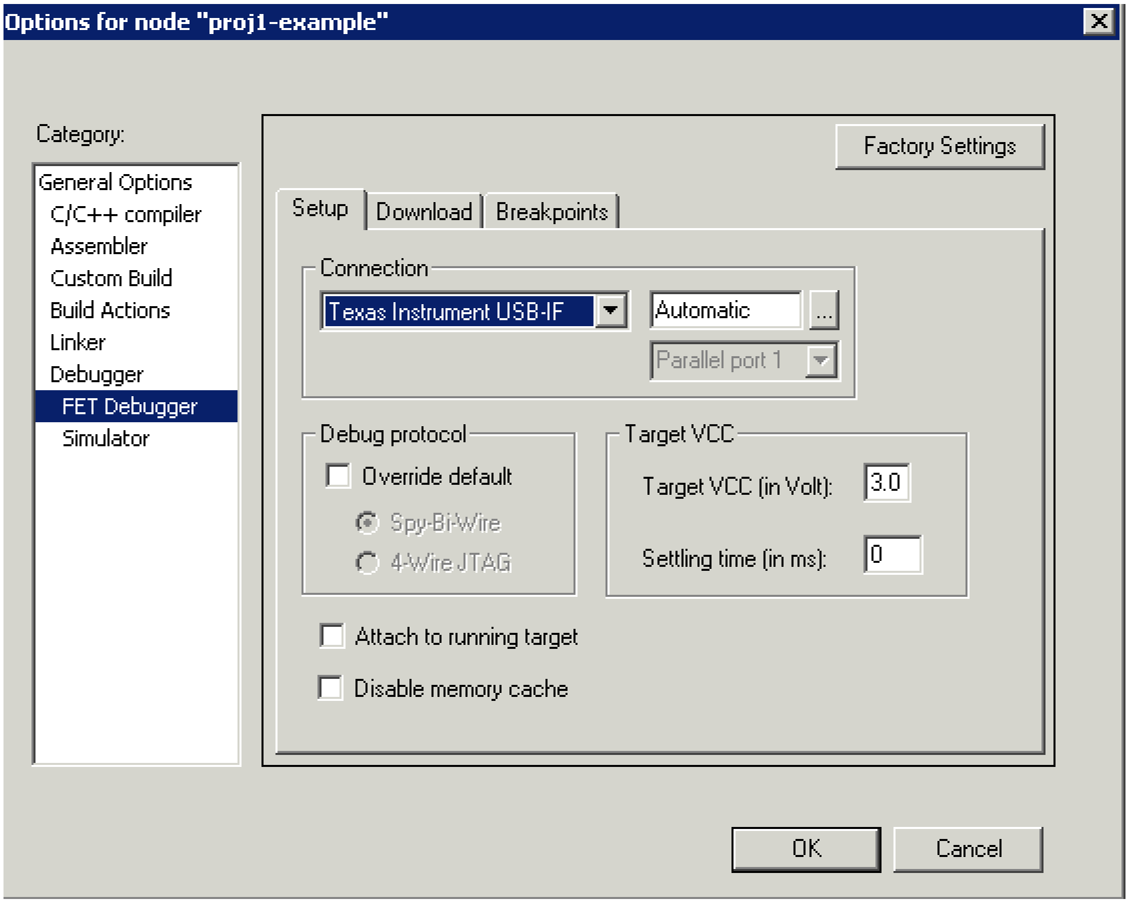
Device description file

□ Override default

$TOOLKIT_DIR$\config\MSP430F2013.ddf

OK Cancel

Step 8: Select **FET Debugger** in the Category section and make the following selections for **Setup** and **Download** tabs.



Options for node "proj1-example"

Category:

- General Options
- C/C++ compiler
- Assembler
- Custom Build
- Build Actions
- Linker
- Debugger
- FET Debugger
- Simulator

Factory Settings

Setup | Download | Breakpoints

Connection: Texas Instrument USB-IF; Automatic; Parallel port 1

Debug protocol

□ Override default

● Spy-Bi-Wire

○ 4-Wire JTAG

Target VCC

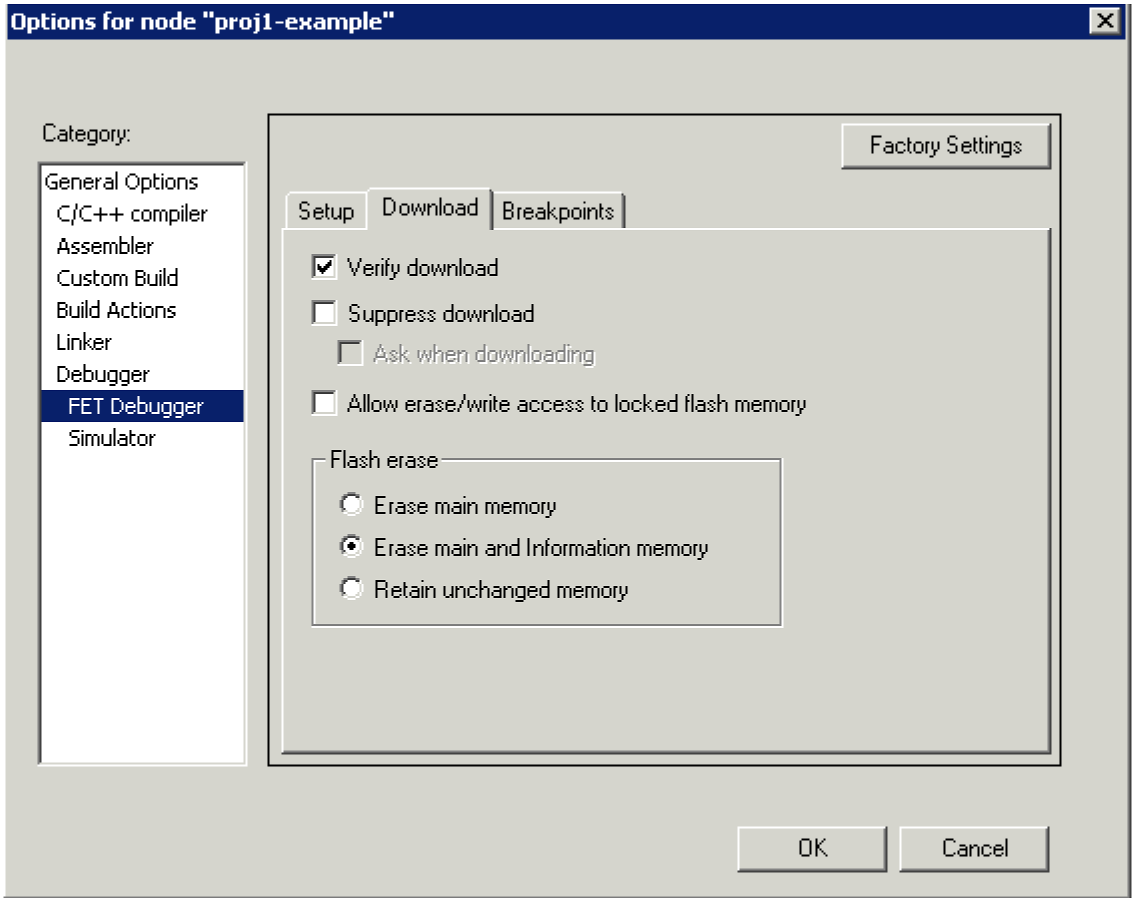
Target VCC (in Volt): 3.0

Settling time (in ms): 0

□ Attach to running target

□ Disable memory cache

OK Cancel



Options for node "proj1-example"

Category:

- General Options
- C/C++ compiler
- Assembler
- Custom Build
- Build Actions
- Linker
- Debugger
  - FET Debugger
  - Simulator

☑ Verify download

□ Suppress download

□ Ask when downloading

□ Allow erase/write access to locked flash memory

Flash erase

○ Erase main memory

● Erase main and Information memory

○ Retain unchanged memory

Click 'OK' when completed.

You are now ready to enter your code. The code will be saved in the 'main.c' file.

Step 9: Using your mouse, highlight the following text (program), beginning at **START HERE** and ending at **END HERE**. Select 'Edit->Copy'.

In the Workbench IDE, highlight the text in the 'main.c' window. Select 'edit->paste'. You have just replaced all the text in the 'main.c' window with the program you copied from this web page.

```
// START HERE
//******************************************************************************
//  MSP430x2xx Demo - Software Toggle P1.0
//
//  Description; Toggle P1.0 by xor'ing P1.0 inside of a software loop.
//  ACLK = n/a, MCLK = SMCLK = default DCO
//
//                MSP430x2xx
//             -----------------
//         /|\|              XIN|-
//          | |                 |
//          --|RST          XOUT|-
//            |                 |
//            |             P1.0|-->LED
//
//  A. Dannenberg
//  Texas Instruments, Inc
//  January 2006
//  Built with IAR Embedded Workbench Version: 3.40A
//******************************************************************************

#include "msp430.h"
```

```
int main(void)
{
  WDTCTL = WDTPW + WDTHOLD;              // Stop watchdog timer
  P1DIR |= 0x01;                         // Set P1.0 to output direction

  for (;;)
  {
    volatile unsigned int i;             // volatile to prevent optimization

    P1OUT ^= 0x01;                       // Toggle P1.0 using exclusive-OR

    i = 10000;                           // SW Delay
    do i--;
    while (i != 0);
  }
}
// END HERE
```

What does this program do?

The above program blinks the LED (light emitting diode) on the eZ430-F2013. The code may look very strange to you, but it is actually quite simple. When you write source code, you should always be sure to carefully comment your code. In C, comments begin with a ***//*** on a line. All text that appears to the right of the ***//*** is considered comments. The compiler will ignore that text.

Every C program must have a function named **main**. This lets the operating system know where to start executing your code. This program is very simple, it only has one function, and that function is named **main**.

Within the main function is a **for** loop. There are no values in the **for** loop to get it to stop, therefore, it will execute forever. Many embedded systems programs run in this way. Since they have one purpose in life, they start running and simply continue.

Inside the **for** loop is another loop that is controlled by the variable **i**. The program stays in a *tight loop*, counting down from the value **i** to 0. This is called a software delay. Each time the counter gets to **0**, the software toggles the light (e.g. if it is on, it turns it off) and starts counting down again.

Step 10: In the Workbench IDE, select 'File->Save Workspace'

You are now ready to compile, download and run your program.

Resources

- **Connexions: Introduction to the Texas Instruments ez430** (Reference material presented here. Some material and screen shots were replicated from this site.)

HOME Running Your Code>>