MSP430FR4133 LaunchPad

Project0

Texas Instruments

Overview

In this exercise you will learn:

- How to create a new project with Code Composer Studio (CCS)
- Learn how to blink the on-board LED2 on the MSP430 LaunchPad
- Change the speed of the blinking LED2
- Learn how to toggle between the LED1 and LED2

Things you will need:

- MSP430FR4133 LaunchPad Development Kit
  - MSP-EXP430FR4133
- Code Composer Studio (Integrated Development Environment)
- 10-15 minutes

Hardware Setup

- The MSP430FR4133 LaunchPad Development Kit includes everything you need out of the box

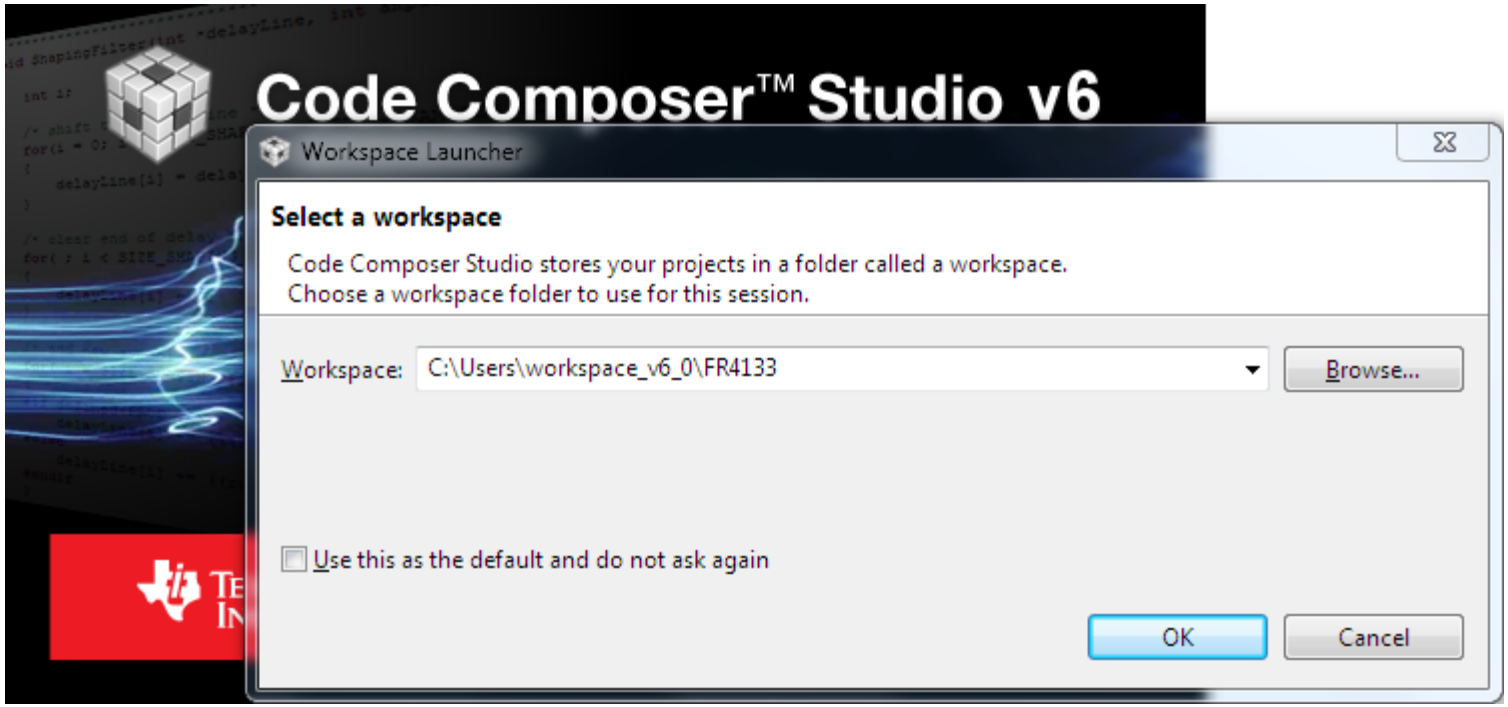
- To start programming, you'll first have to install CCS which contains the required drivers for your new MSP430FR4133 LaunchPad

- Plug your FR4133 LaunchPad into the PC with the included USB cable

- If prompted, let Windows automatically install the software.

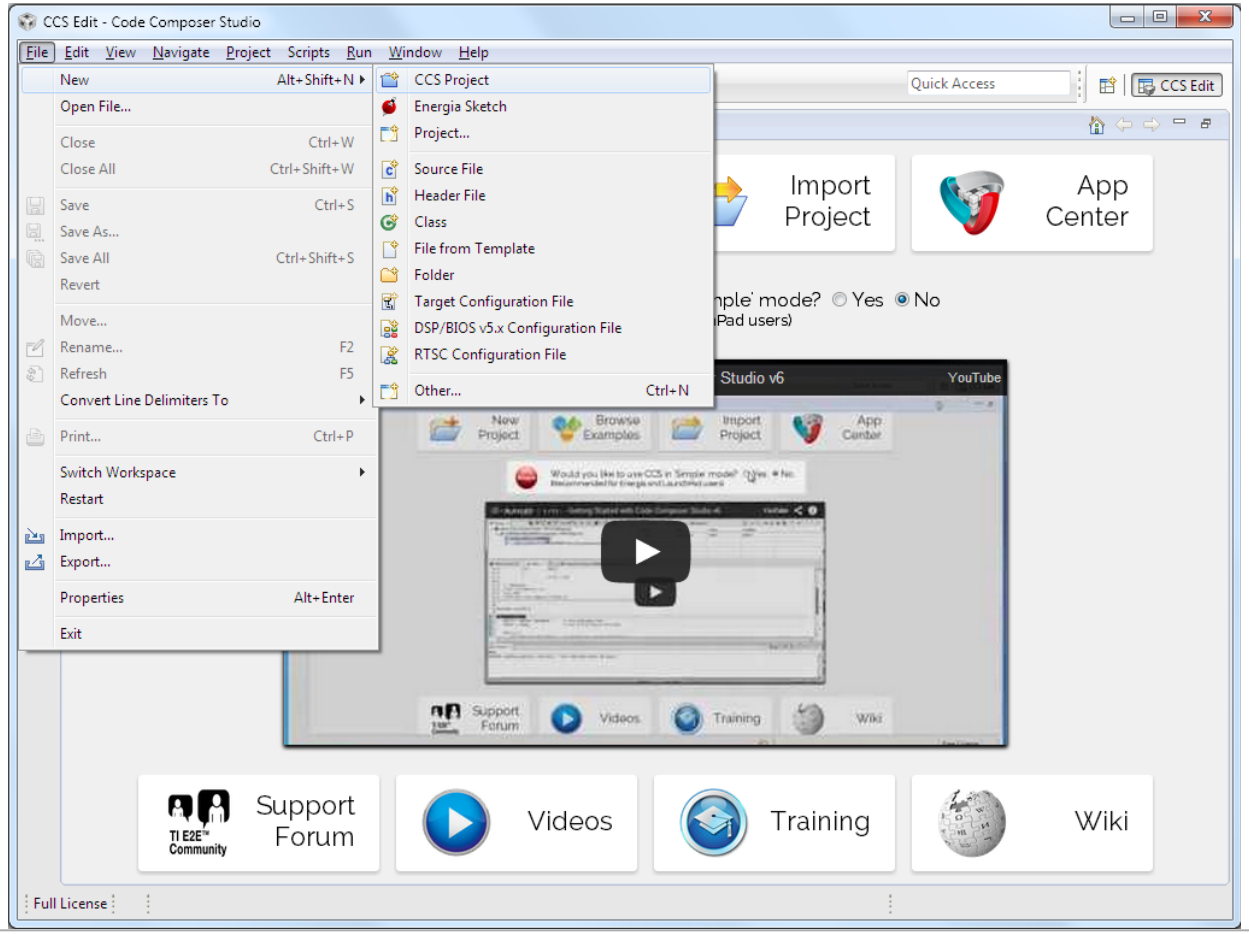
Create a New CCS Workspace

- Upon opening CCS, it will ask you to select a workspace
- A workspace is where all of your CCS projects will be stored. Create a workspace folder, then *press OK*



Create a New CCS Project

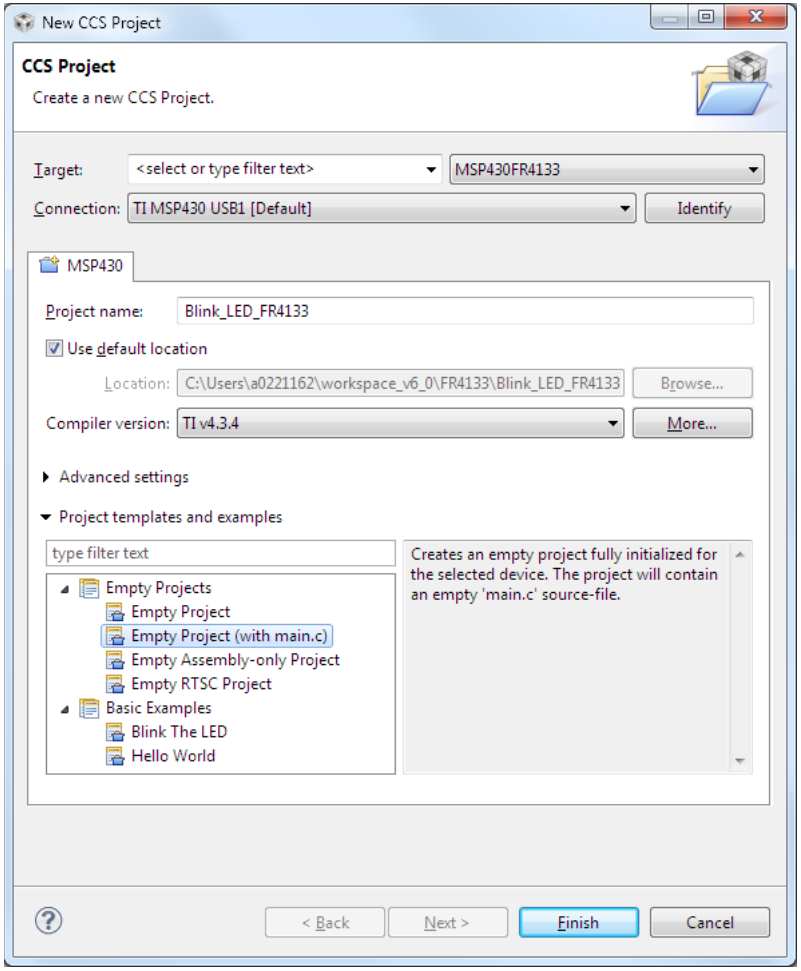
- Once CCS is opened, we can create a new project by going to to *File* > *New* > *CCS Project*



Choose CCS Project Settings

This will open up the "New CCS Project" window. Within this window, we need to do 2 things.

1. Choose our *Target device*
2. Name our project



1. We need to choose the appropriate MSP430 device. For this tutorial, we will program the MSP430FR4133 device that comes pre-populated on the MSP430FR4133 LaunchPad
2. Let's name our project "Blink_LED_FR4133“

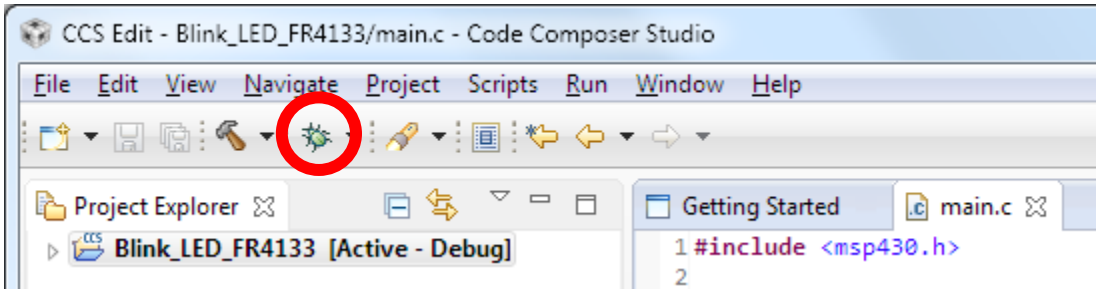
Write Code

In the main.c file we have a blank canvas to write our code!

```
#include <msp430.h>

int main(void)
{
  WDTCTL = WDTPW + WDTHOLD;  // Stop Watchdog Timer. This line of code is needed at the beginning of most MSP430 projects
                             //This turns off the watchdog timer, which can reset the device after a certain period of time

  // Configure GPIO
  P4DIR |= 0x01;             // P4DIR is a register that configures the direction (DIR) or a port pin as an output or an input

                             // To set a specific pin as output or input, we write a '1' or '0' on the appropriate bit of the register.
                             // P4DIR = <PIN7><PIN6><PIN5><PIN4><PIN3><PIN2><PIN1><PIN0>
                             // Since we want to blink the LED2, we want to set the direction of Port 4, Pin 0 (P4.0) as an output
                             // We do that by writing a 1 on the PIN0 bit of the P4DIR register
                             // P4DIR = <PIN7><PIN6><PIN5><PIN4><PIN3><PIN2><PIN1><PIN0>
                             // P4DIR = 0000 0001
                             // P4DIR = 0x01 <-- this is the hexadecimal conversion of 0000 0001
                             // The OR command (|) will always set a bit
                             // P4DIR|= 0x01 is equivalent to P4DIR = P4DIR | 0x01
                             // Regardless of value in P4DIR, this OR operation will set Bit0 (or Pin0) to 1-> (1|0 = 1, 1|1 = 1)

  PM5CTL0 &= ~LOCKLPM5;      // Disable the GPIO power-on default high-impedance mode to activate previously configured port settings

  while(1)                   // This while(1) loop will cause the lines of code inside it to loop infinitely
  {
    P4OUT ^= 0x01;           // Toggle P4.0 using exclusive-OR operation (^)
                             // P4OUT is another register which holds the status of the LED.
                             // '1' specifies that it's ON or HIGH, while '0' specifies that it's OFF or LOW
                             // Since our LED is tied to P4.0, we will toggle the 0 bit of the P4OUT register
                             // The exclusive-OR command (^) will always toggle a bit at a certain position
                             // P4OUT^= 0x01 is equivalent to P4DIR = P4DIR ^ 0x01
                             // Regardless of value in P4OUT, this OR operation will toggle Bit0 (or Pin0) (1^0 = 1, 1^1 = 0)

    __delay_cycles(1000000);
  }
}
```

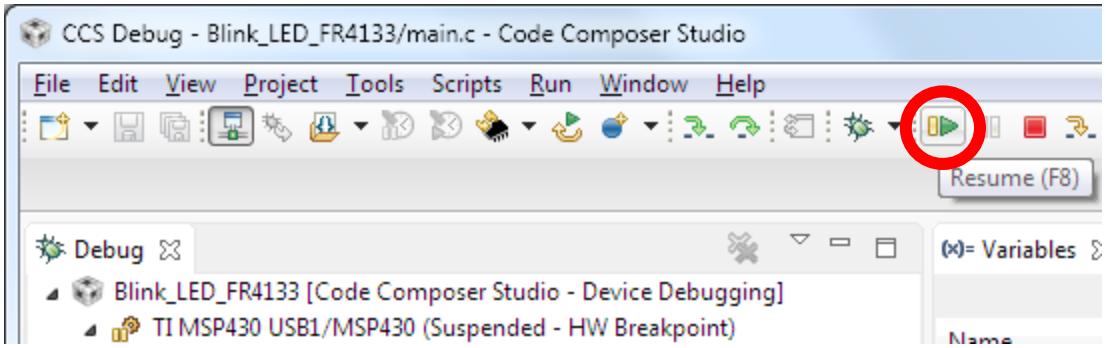
Download Code

- Now that we have written our code, we can download it to our MSP430 LaunchPad that is plugged into the USB port!
- Code is downloaded by clicking the *debug* button



Run Code

- Clicking the Debug button will take us to the CCS Debug View. Here, we can *press the Run button* to start running the code we just wrote.



- At this point, your green LED should be blinking. *Congratulations!*

Additional Exercises

- Now that we have our LED2 blinking, play around with the number inside of the while-loop to change the speed of the blinking LED. The smaller the number, the shorter the delay between LED toggles, or the faster the blinking. Alternatively, the larger the number, the longer the delay. Try values like 5000, 40000, etc.

- Another exercise is getting the LED1 to blink as well. The LED1 is tied to Port P1.0. Using the P1DIR and P1OUT registers along with P4DIR and P4OUT we used above, see if you can get both LEDs to blink. Can you make them blink in unison? Can you make them blink alternatively?

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265