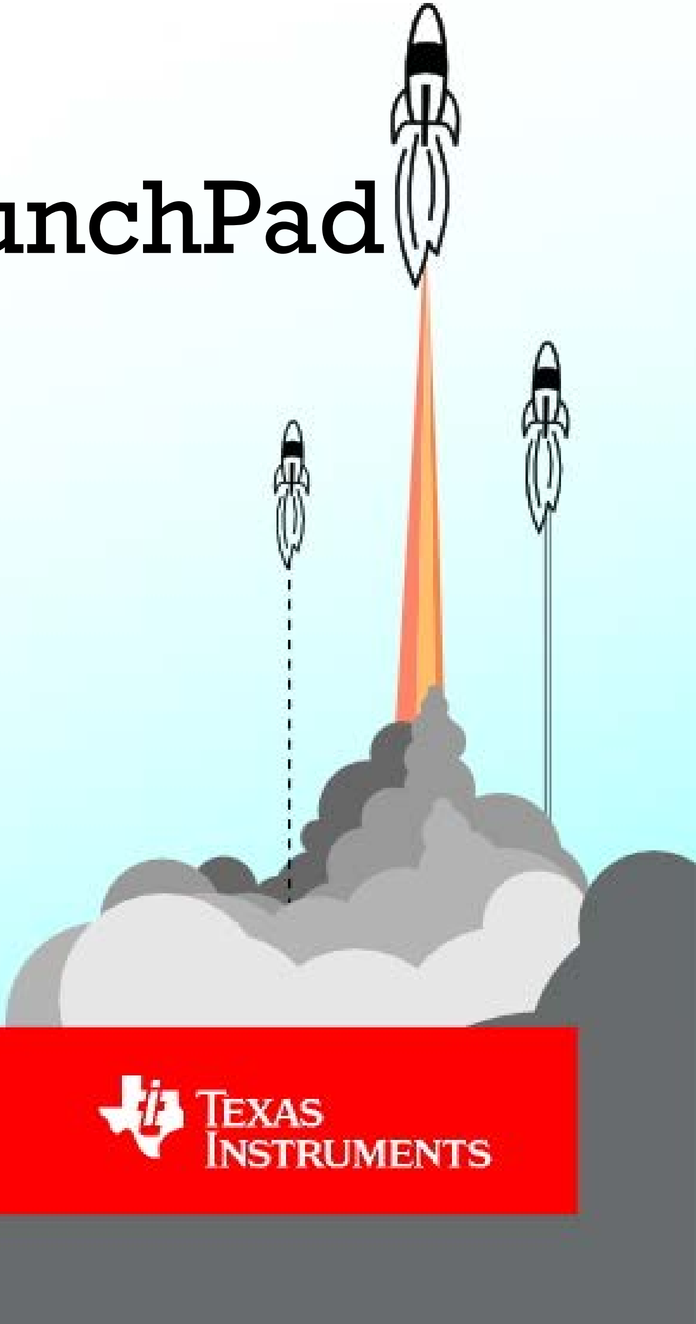
MSP430FR4133 LaunchPad

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:

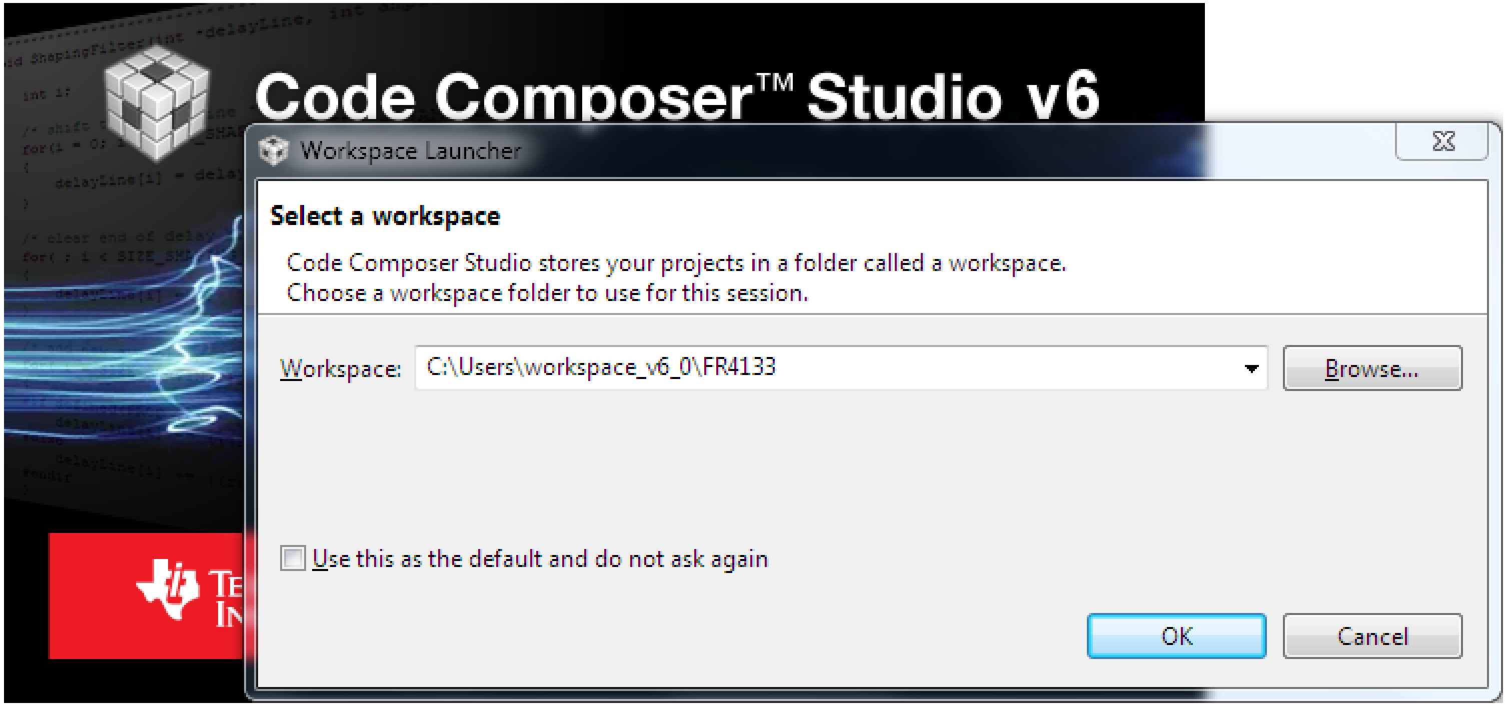
- MSP430FR4133 LaunchPad Development Kit
  - MSP-EXP430FR4133
- Code Composer Studio (Integrated Development Environment)
- 10-15 minutes

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*



Select a workspace

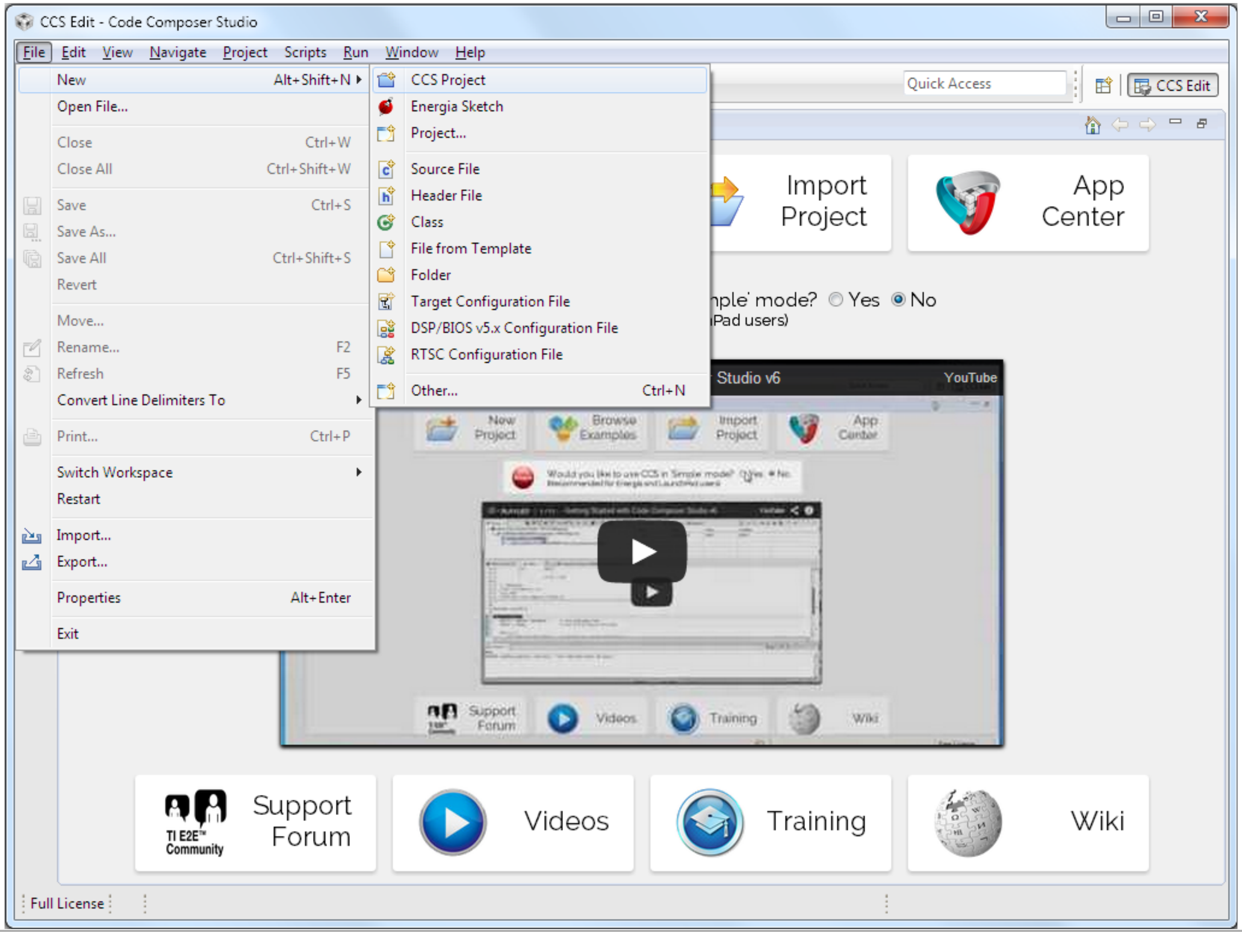
Code Composer Studio stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

Workspace: C:\Users\workspace_v6_0\FR4133

□ Use this as the default and do not ask again

Create a New CCS Project

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

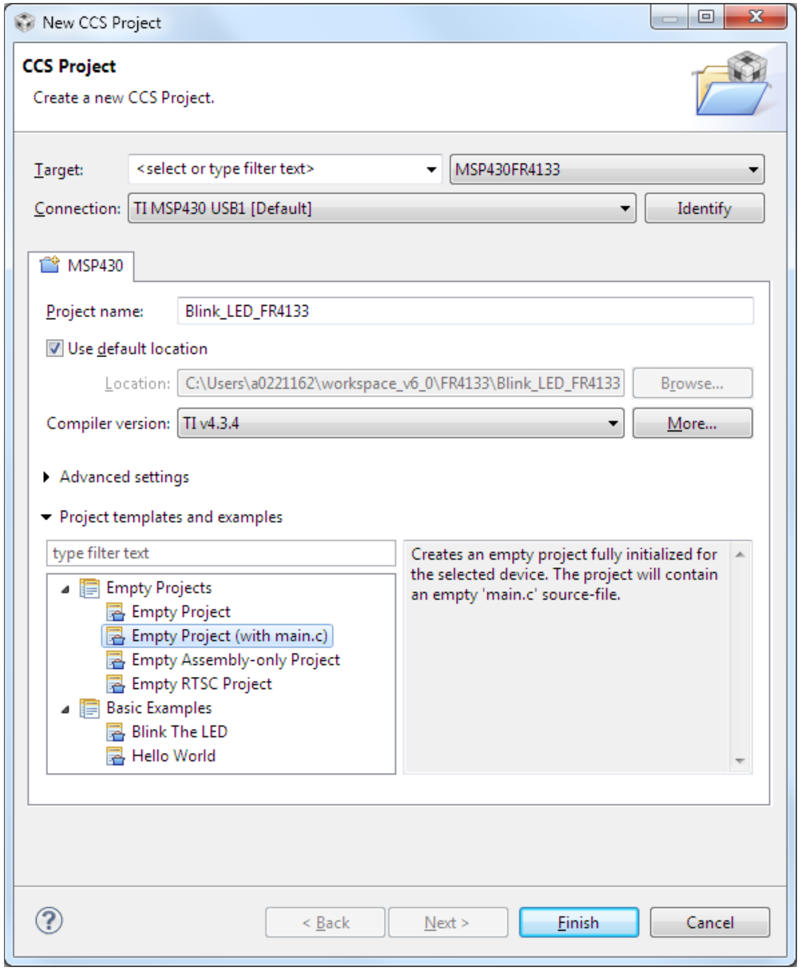


Choose CCS Project Settings

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“



New CCS Project

CCS Project

Create a new CCS Project.

Target: <select or type filter text> MSP430FR4133

Connection: TI MSP430 USB1 [Default]

Project name: Blink_LED_FR4133

☑ Use default location

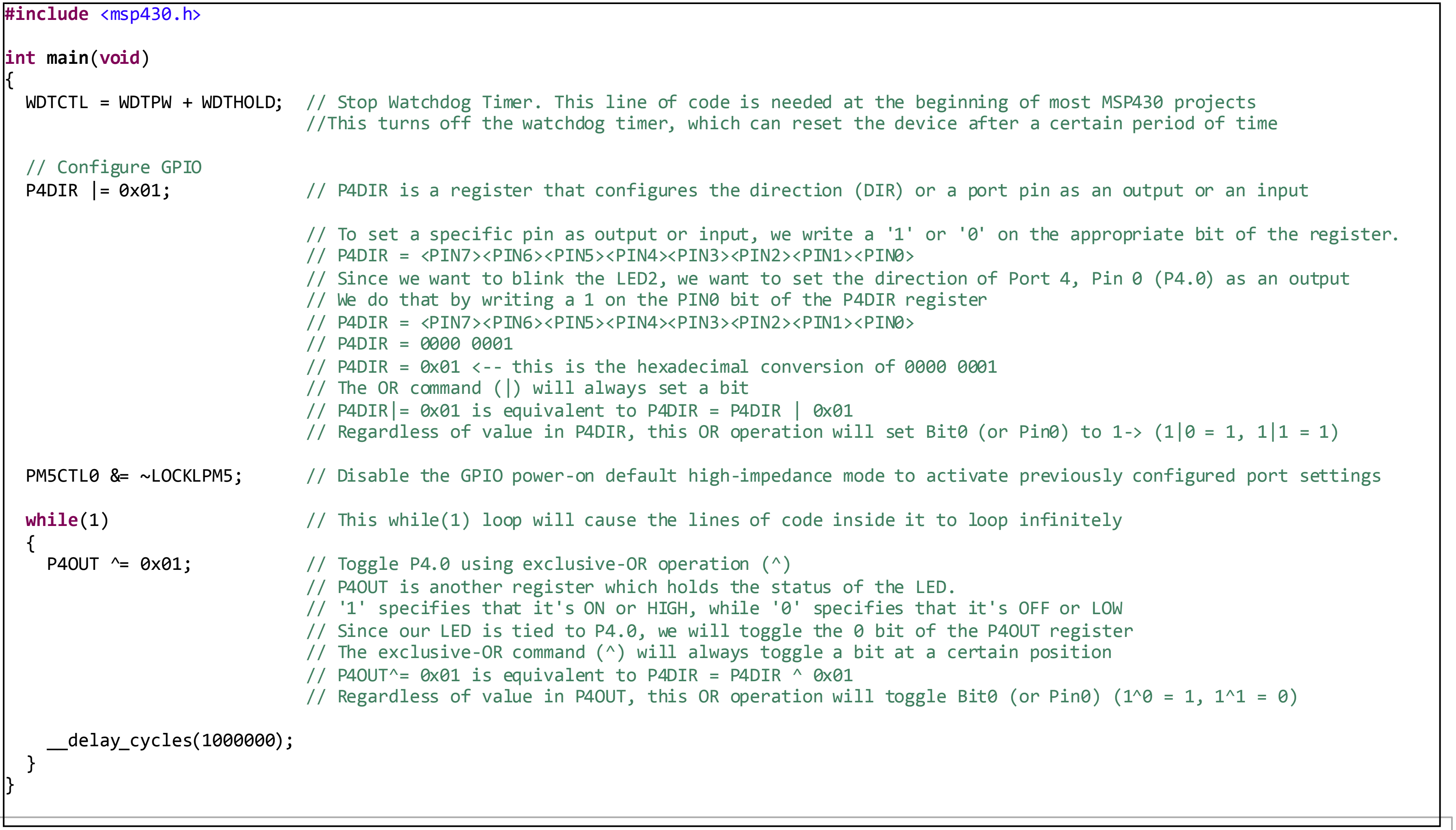
Location: C:\Users\a0221162\workspace_v6_0\FR4133\Blink_LED_FR4133

Compiler version: TI v4.3.4

Creates an empty project fully initialized for the selected device. The project will contain an empty 'main.c' source-file.

Write Code

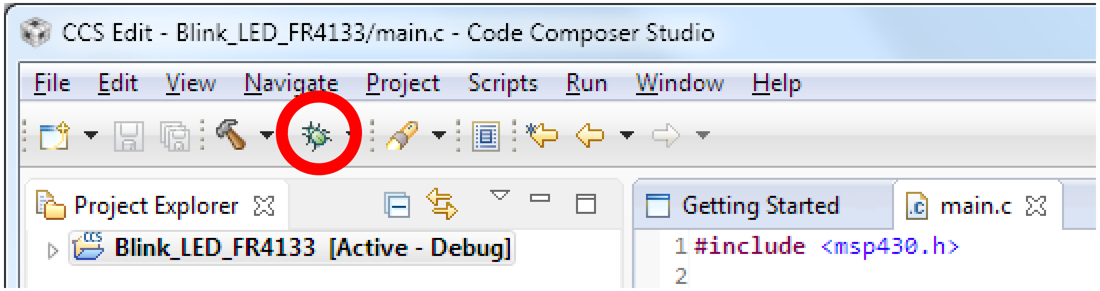
In the main.c file we have a blank canvas to write our code!



```c
#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

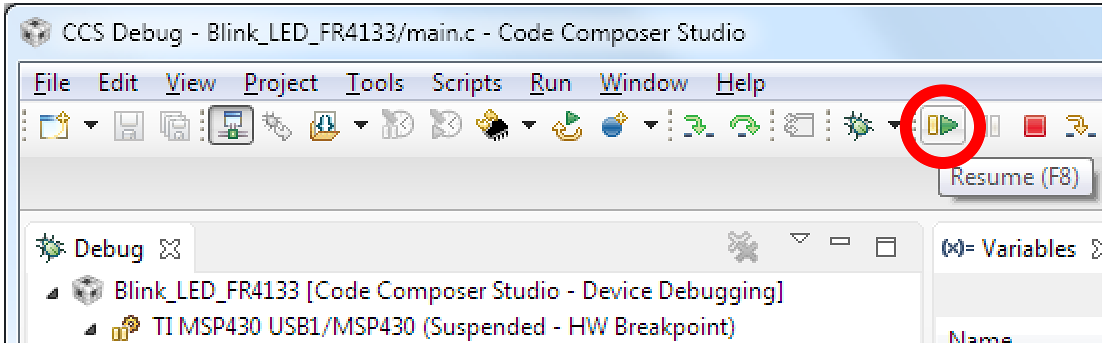
- Now that we have written our code, we can download it to our MSP430 LaunchPad that is plugged into the USB port!
- Code is downloaded by clicking the *debug* button


CCS Edit - Blink_LED_FR4133/main.c - Code Composer Studio
File Edit View Navigate Project Scripts Run Window Help
Project Explorer
Blink_LED_FR4133 [Active - Debug]
Getting Started
main.c

```
#include <msp430.h>

```

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.



CCS Debug - Blink_LED_FR4133/main.c - Code Composer Studio

Blink_LED_FR4133 [Code Composer Studio - Device Debugging]

TI MSP430 USB1/MSP430 (Suspended - HW Breakpoint)

- 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

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as “components”) are sold subject to TI’s terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI’s terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers’ products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers’ products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of significant portions of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI’s goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or “enhanced plastic” are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have ***not*** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

Products		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		

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