

CODE COMPOSER STUDIO (CCS) TUTORIAL

Created by Phil Bones, 2018

Updated by Ciaran Moore and Le Yang, 2019, 2020, 2021 and 2022

For CCS 10.1.1

Starting your first ENCE361 lab with CCS (1)

- On all Level 2 lab PCs (in Electronics, Embedded Systems and Computer Labs), CCS 10.1.1/11.1.0 has been installed
- The API for the TM4C series microcontroller is called TivaWare. It is installed in **C:\ti\TivaWare_C_Series-2.2.0.295**
- During labs, use CCS and link with the TivaWare API on the local PC with your source code and projects on your **P: drive**
- **If you are setting up CCS on your laptop, you will need to install CCS first and then TivaWare**
- This guide gets you started. **You may need to read it again for later labs**

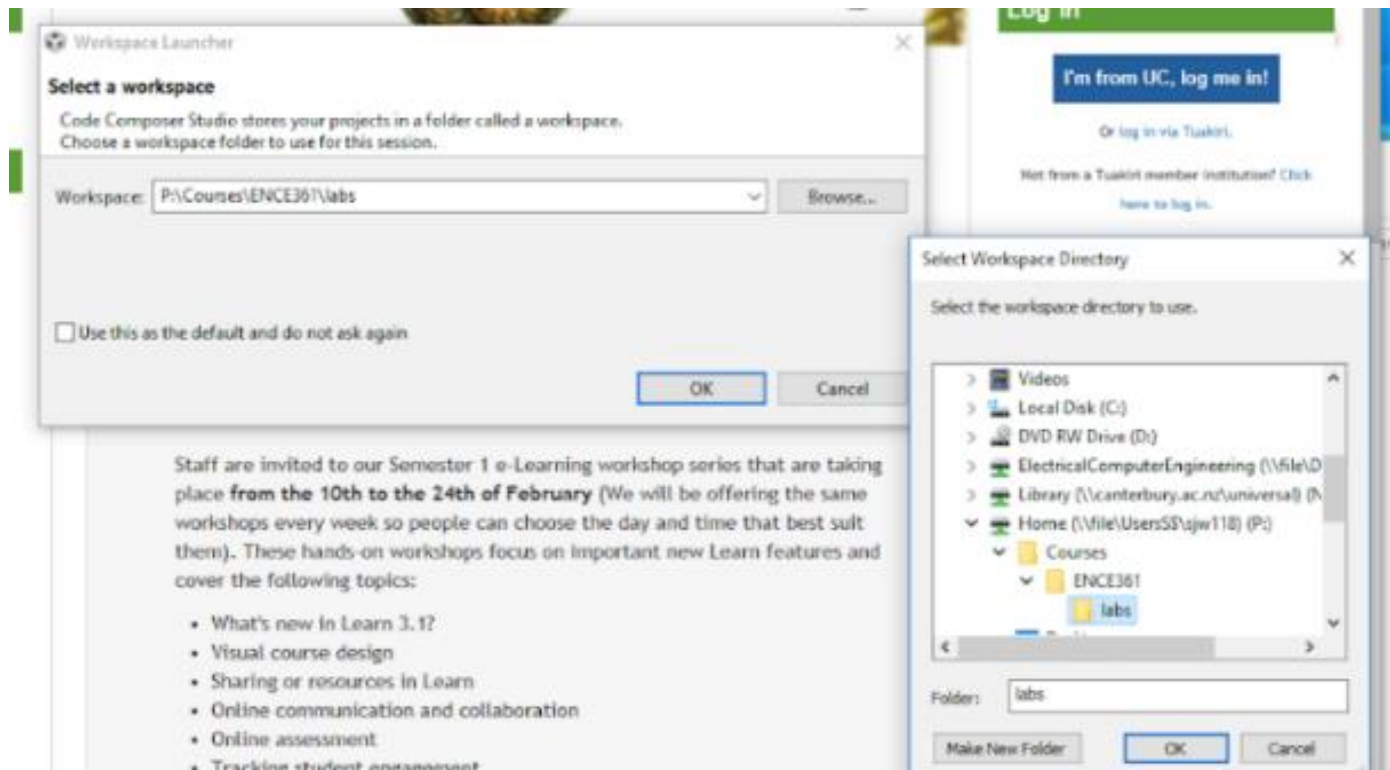
Starting your first ENCE361 lab with CCS (2)

During the lab session in Week-2, **you will:**

- i. Learn to configure CCS project using a simple GPIO example
- ii. Learn how to [compile](#), [link](#) and [debug](#) a CCS project using TM4C123GH6PM microcontroller
- iii. Configure and use application the programming interface (API), TivaWare
- iv. Understand how general purpose input and output (GPIO) work on TM4C series microcontroller by examining and modifying a C source code

Log in and setting a CCS workspace on P: drive

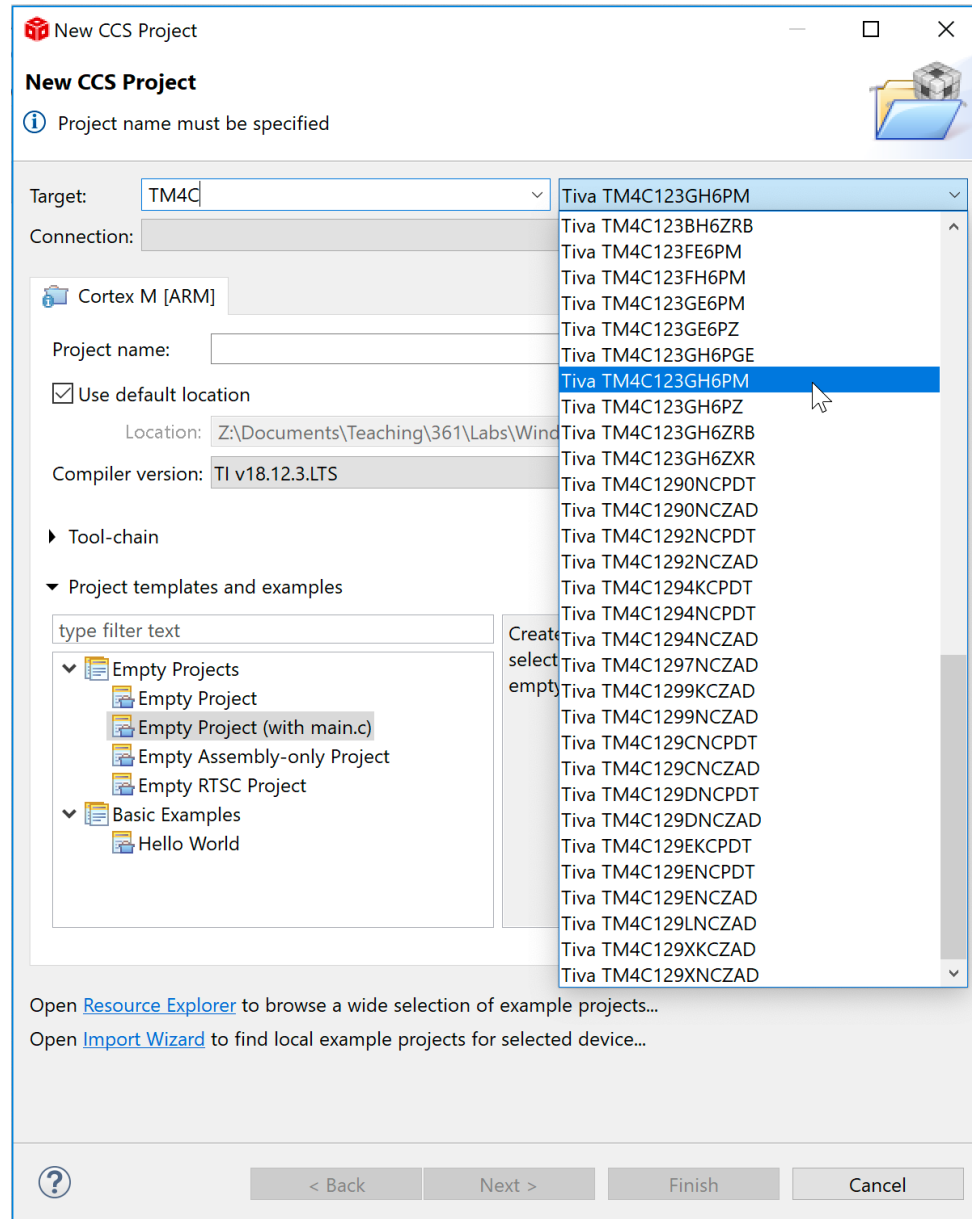
- Login to a PC in the Electronics lab with your UOCNT account
- Make sure your P: drive is mapped. Check this by opening a file explorer window, clicking on “This PC” and checking that your usercode is mapped to the “P” drive



- Next, in your workspace, create the directory structure **P: \Courses\ENCE361\labs**
- **In this course, do not use spaces when defining either directory or file names. You will have no end of problems if you use spaces!** Lastly, click OK...

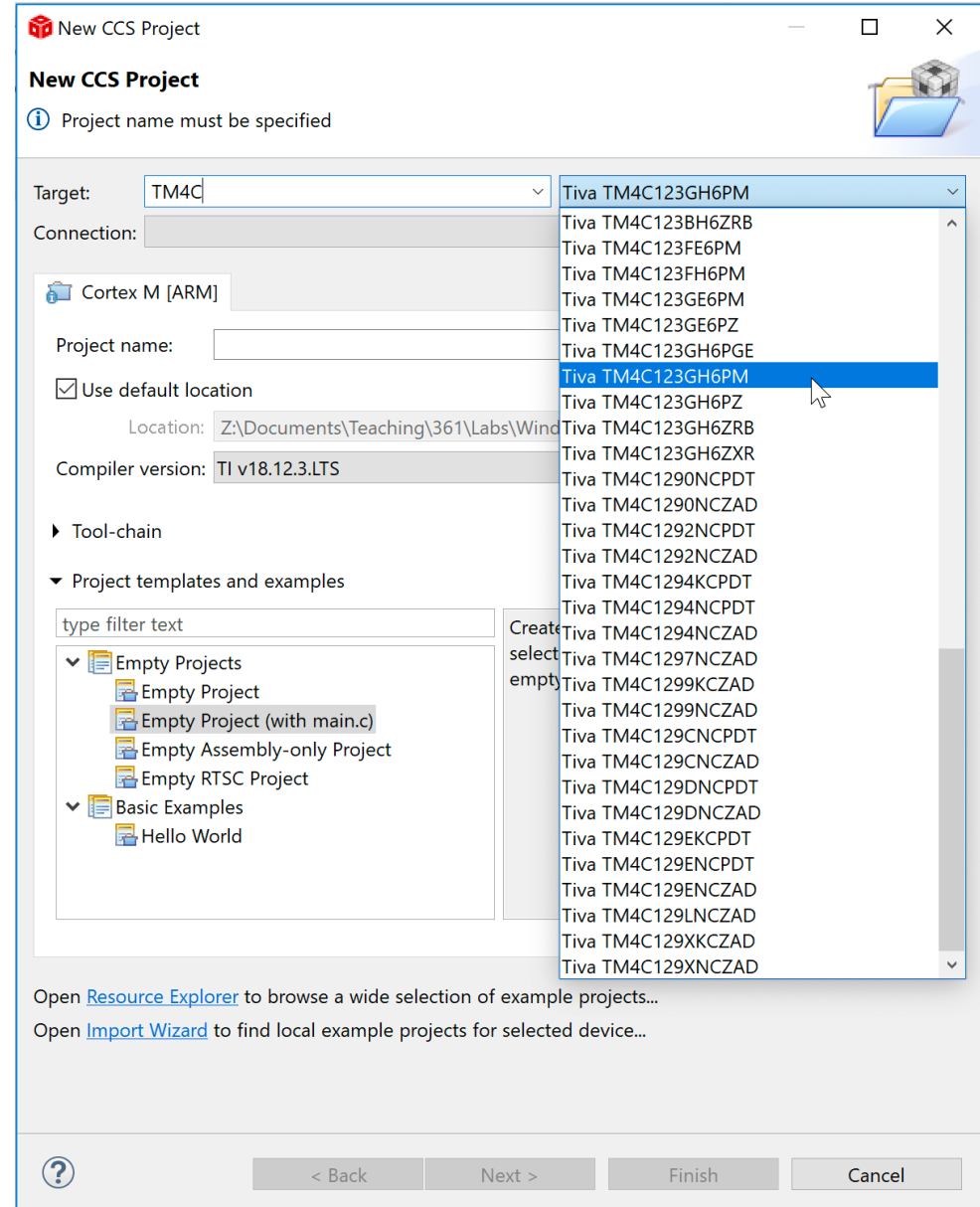
Setting in CCS (1)

- Select File>New>CCS project
- Click on “Target” window shown in the dialog box and start typing ...
“**Tiva TM4C123GH6PM**”

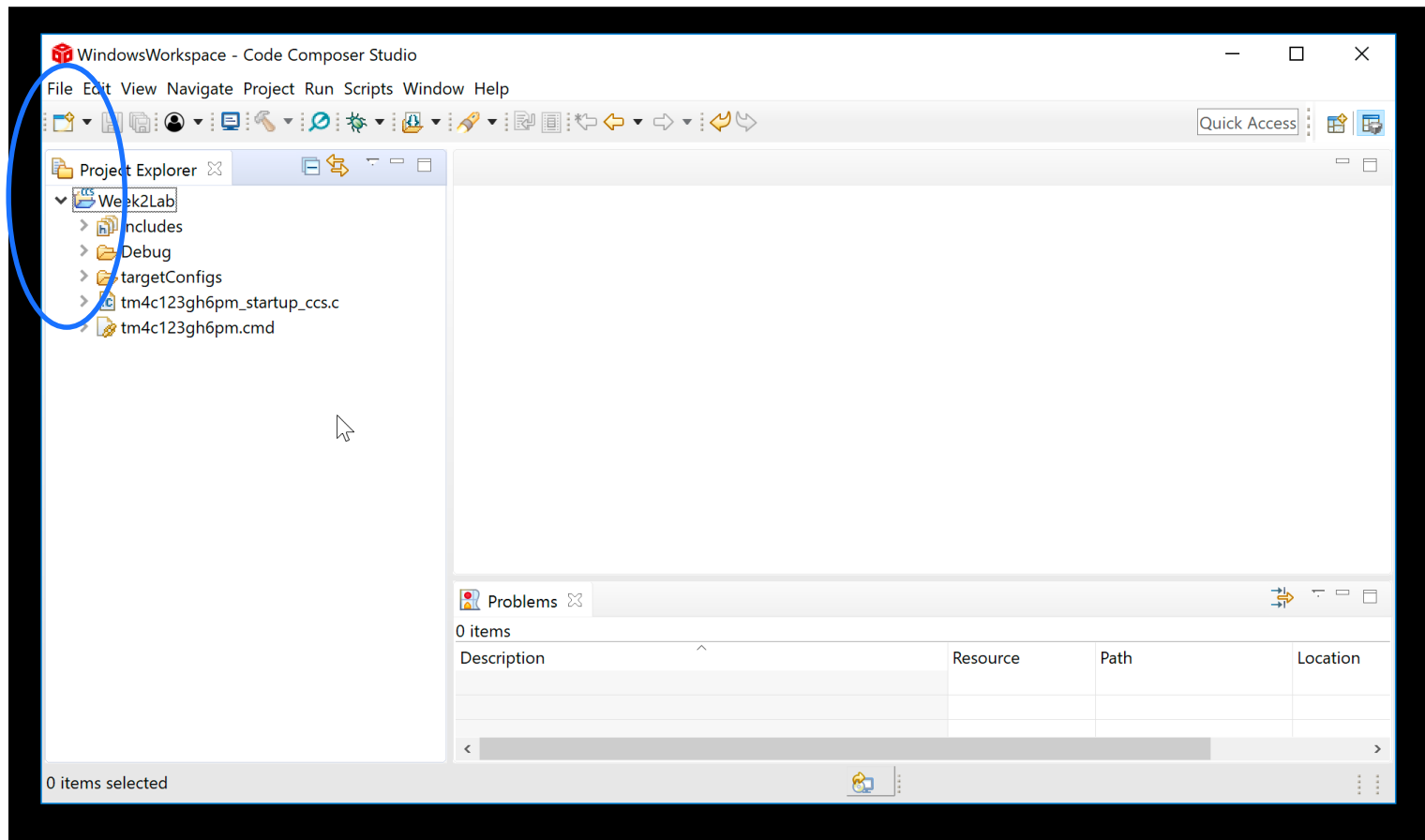


Setting up in CCS (2)

- For “**Project name**”, type in something descriptive, like: “**Week2Lab**”
- For the “**Connection**”, select “**Stellaris in-circuit Debugger Interface**”.
- Lastly, select “**Empty Project**” from the Project “**Templates**” window
- Click ‘**Finish**’



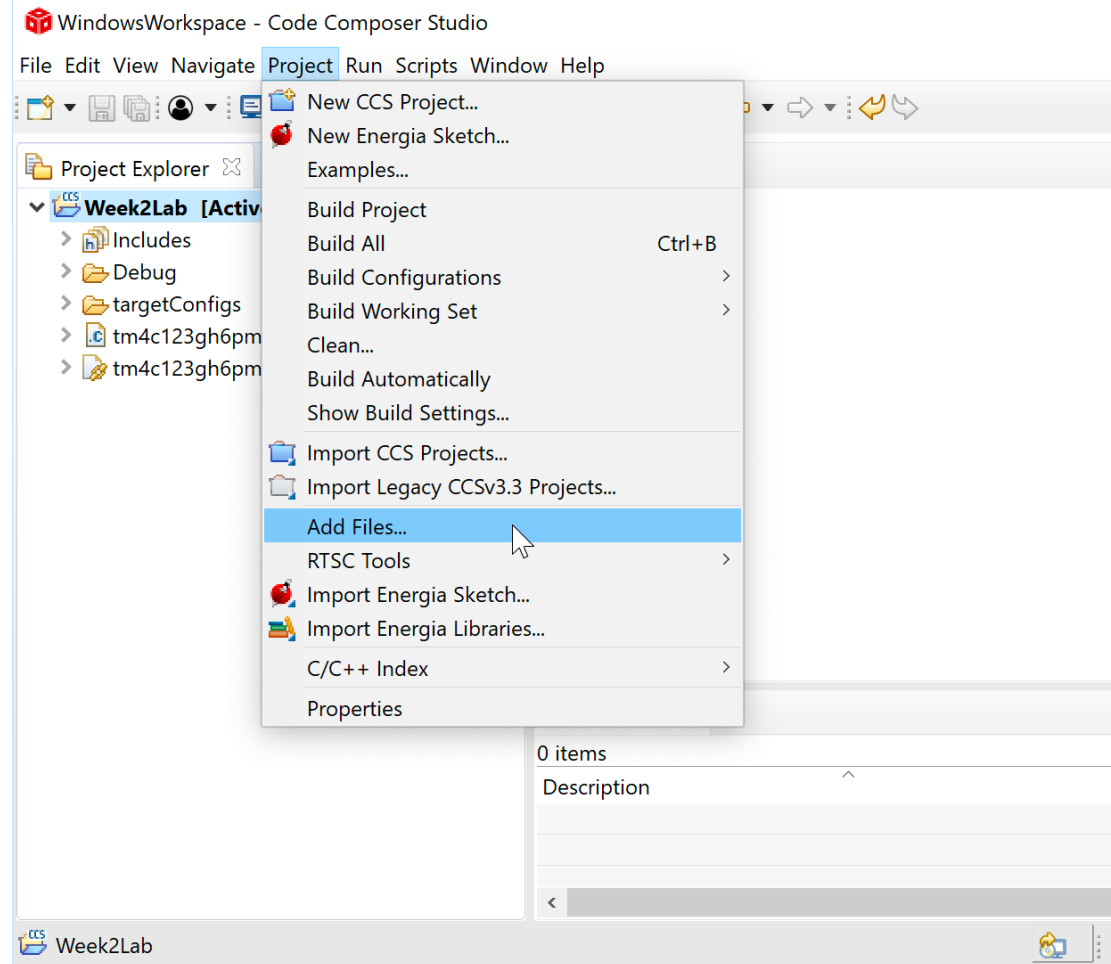
Setting up in CCS (3)



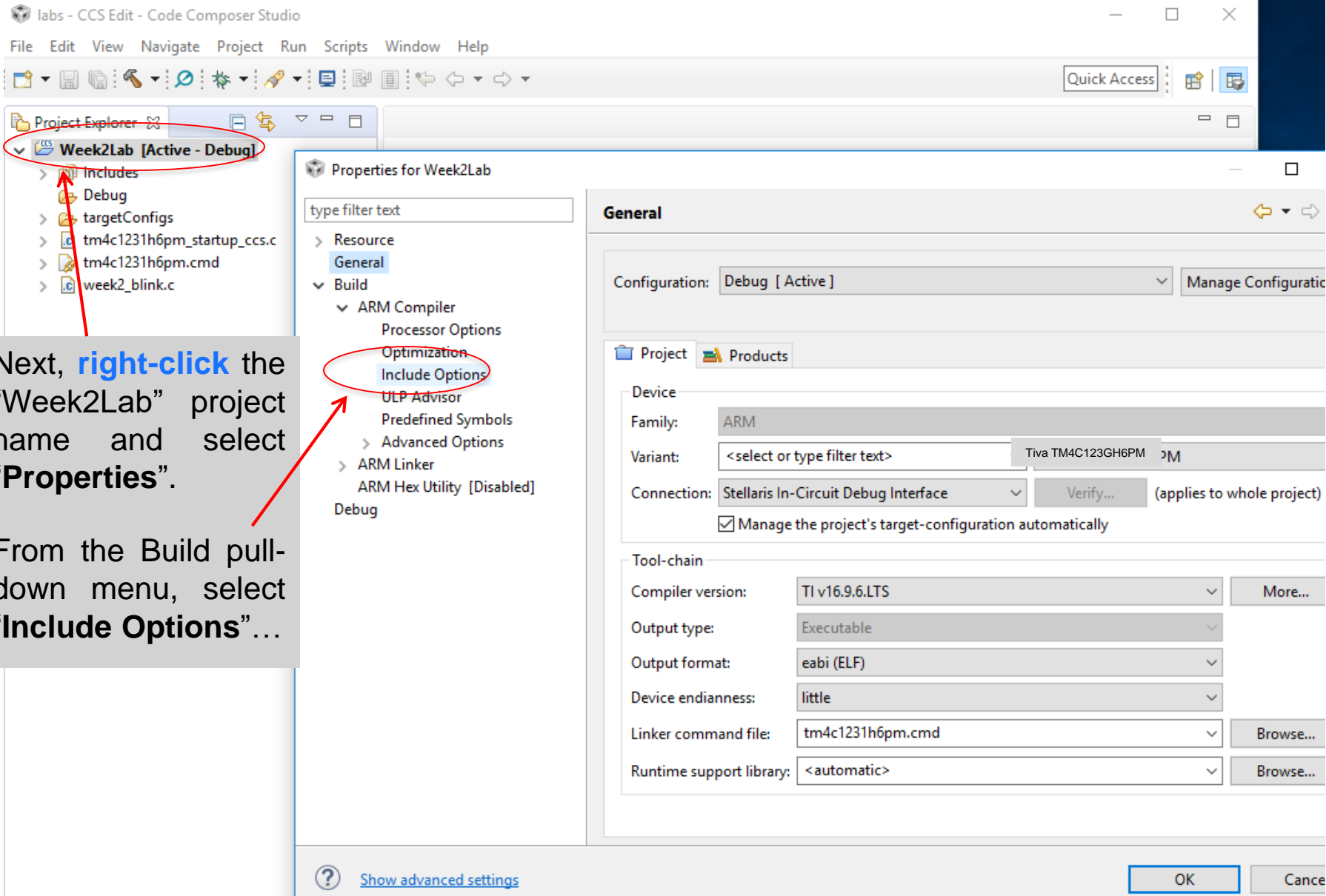
- On the project explorer window, click on the **greater than symbol** to expand the file window and see the initially associated files for the **Week2Lab** project

Setting up in CCS (4)

- For Week 2, there is only one file, **week2_blink.c**
- Copy the source file into **P:\courses\ENCE361\labs\Week2Lab** folder
- The **week2_blink.c** file should have been loaded into your project
- You can also use “Add Files” from the menu, as shown in the right



Setting up in CCS (5)



labs - CCS Edit - Code Composer Studio

File Edit View Navigate Project Run Scripts Window Help

Quick Access

Project Explorer

Week2Lab [Active - Debug]

Includes

Debug

targetConfigs

tm4c1231h6pm_startup_ccs.c

tm4c1231h6pm.cmd

week2_blink.c

Properties for Week2Lab

type filter text

Resource

General

Build

ARM Compiler

Processor Options

Optimization

Include Options

ULP Advisor

Predefined Symbols

Advanced Options

ARM Linker

ARM Hex Utility [Disabled]

Debug

General

Configuration: Debug [Active] Manage Configuration

Project Products

Device

Family: ARM

Variant: <select or type filter text> Tiva TM4C123GH6PM

Connection: Stellaris In-Circuit Debug Interface Verify... (applies to whole project)

☒ Manage the project's target-configuration automatically

Tool-chain

Compiler version: TI v16.9.6.LTS More...

Output type: Executable

Output format: eabi (ELF)

Device endianness: little

Linker command file: tm4c1231h6pm.cmd Browse...

Runtime support library: <automatic> Browse...

Show advanced settings

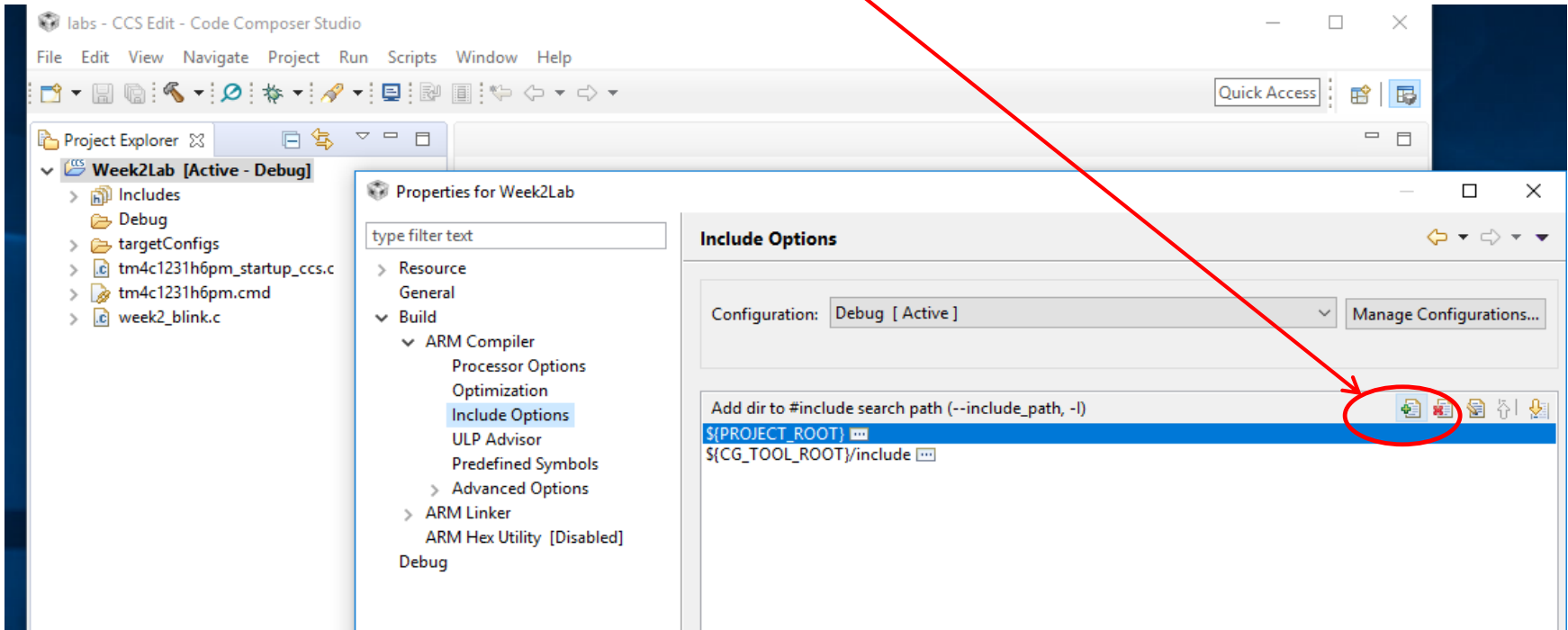
OK Cancel

- Next, **right-click** the “Week2Lab” project name and select **“Properties”**.

- From the Build pull-down menu, select **“Include Options”**...

Setting up in CCS (6)

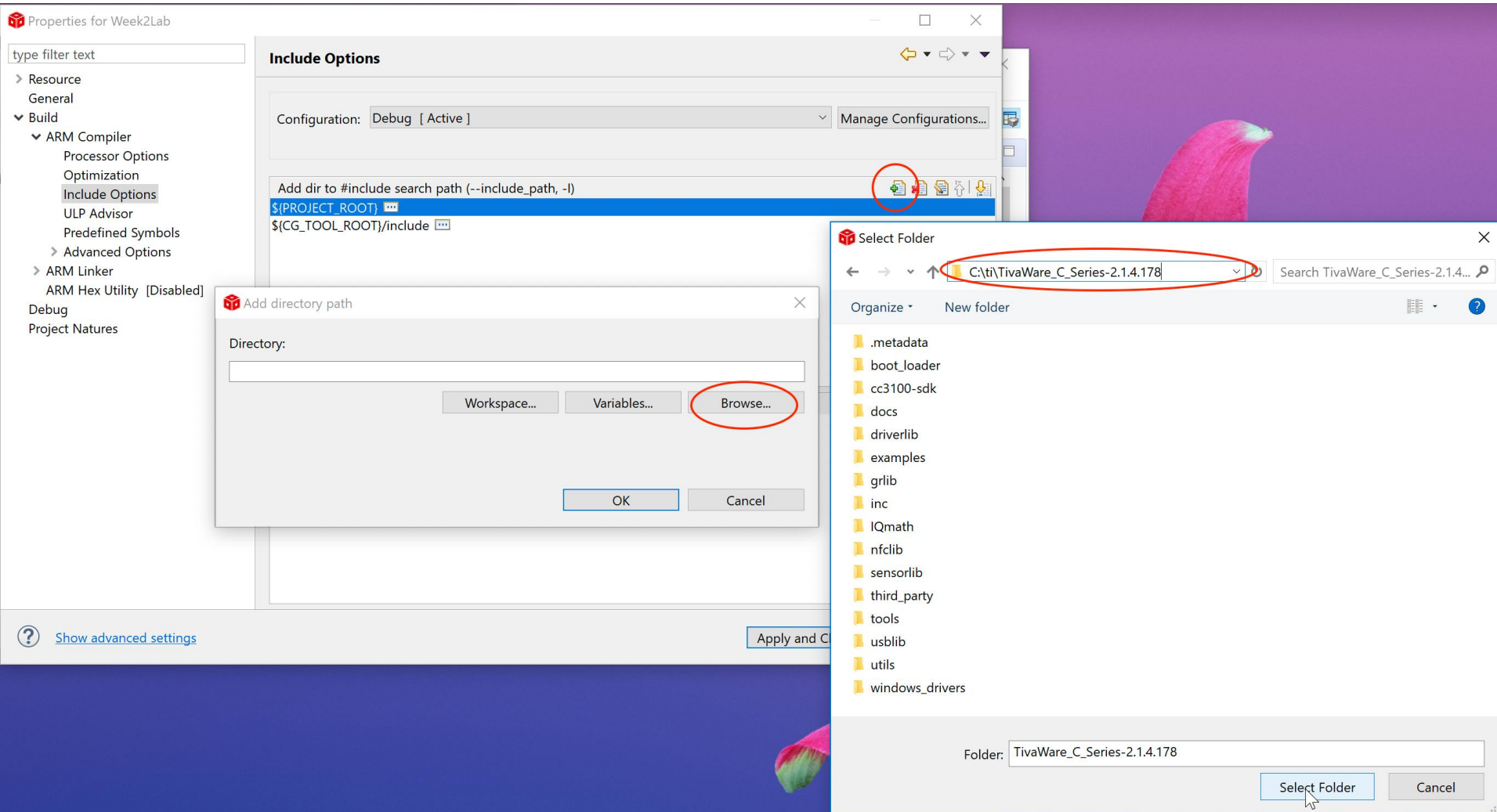
- In the top window, click on the **green '+'** button



Use browse button to find the “**TivaWare_C_Series_2.2.0.295**” directory in C:\ti

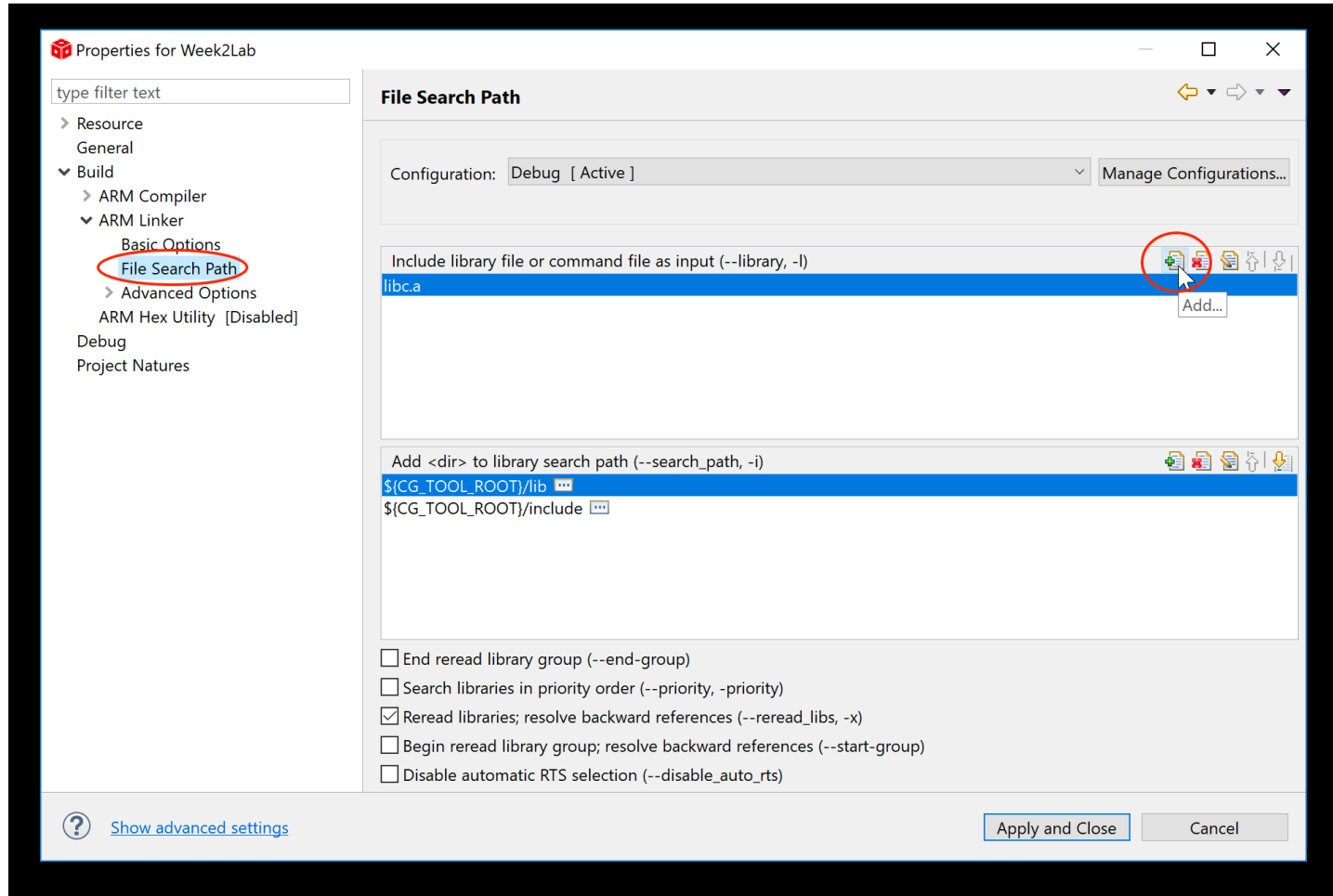
This will allow your CCS compiler to find the include files that CCS requires in order to compile your code

Setting up in CCS (7)



Setting up in CCS (8)

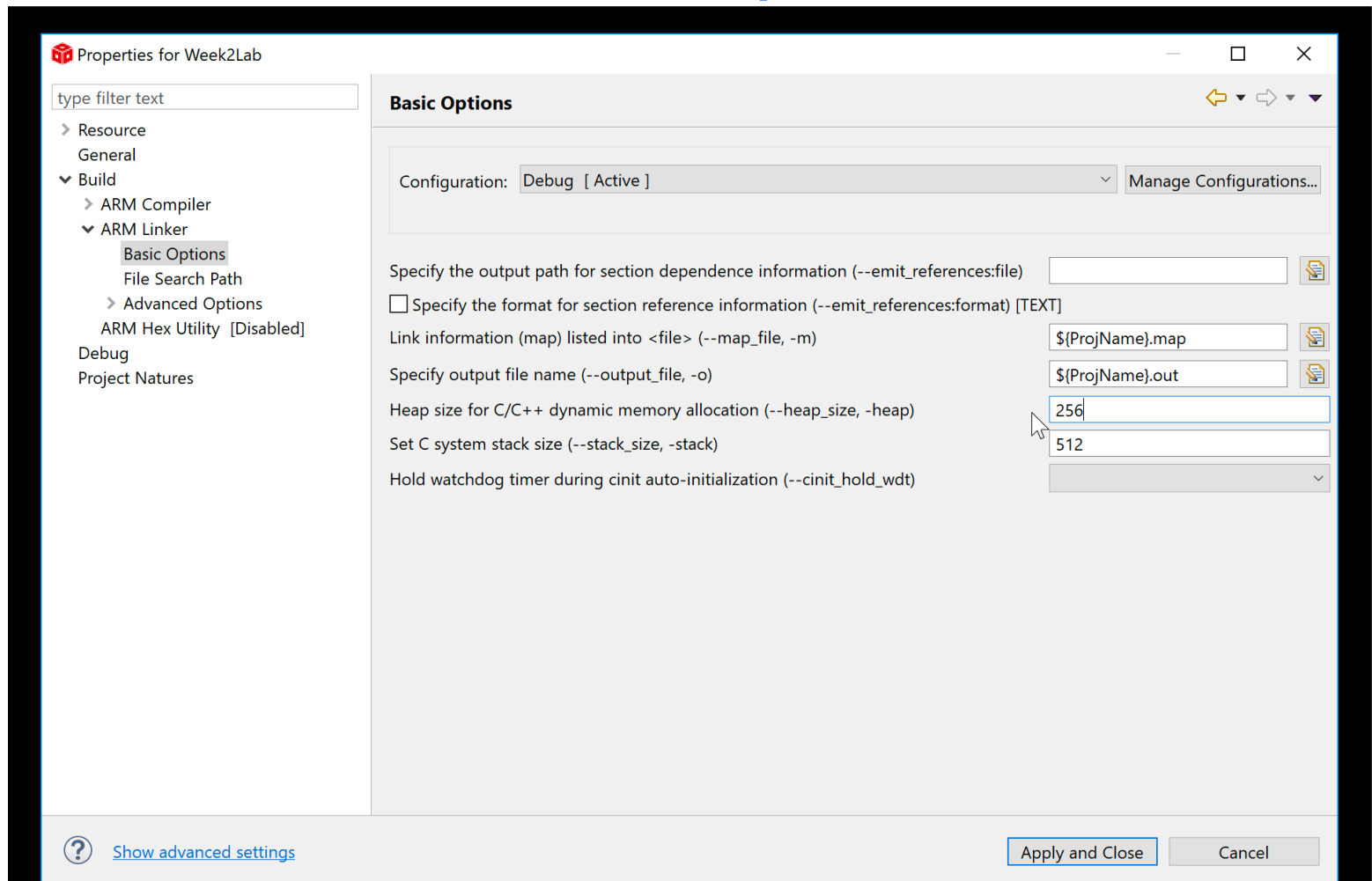
- From the same “**Properties**” dialog box, look down the list on the left side and select “**Build | ARM linker**” and then select “**File search path**”



- Click **green '+' button** in the include library dialog box and select browse button
- Find “**C:\ti\TivaWare_C_Series-2.2.0.295\driverlib\ccs\Debug\driverlib.lib**”

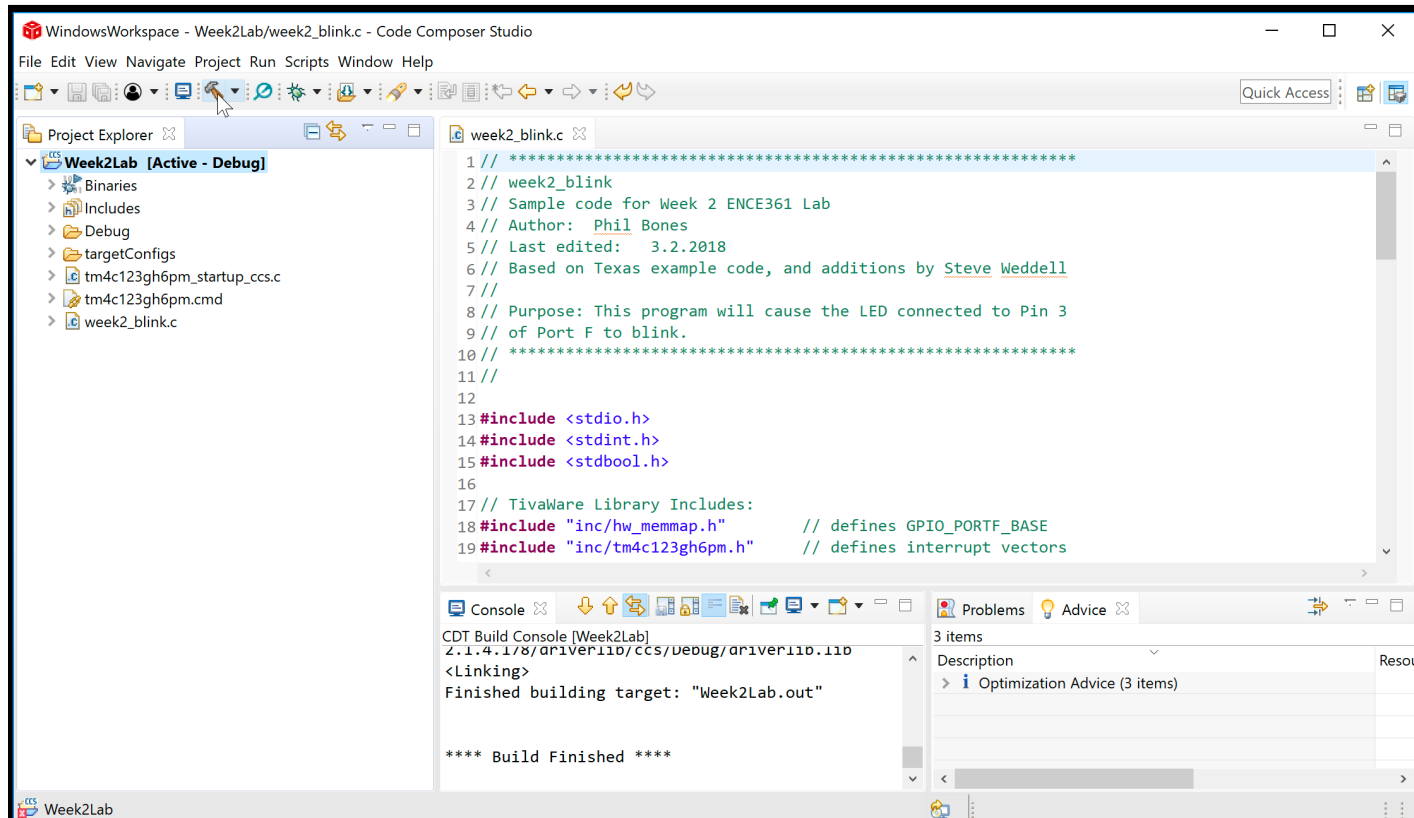
Setting up in CCS (9)

- For some projects, adjusting the heap size and/or stack size is necessary
- Select **Build>ARM Linker>Basic Options**



Compiling and linking

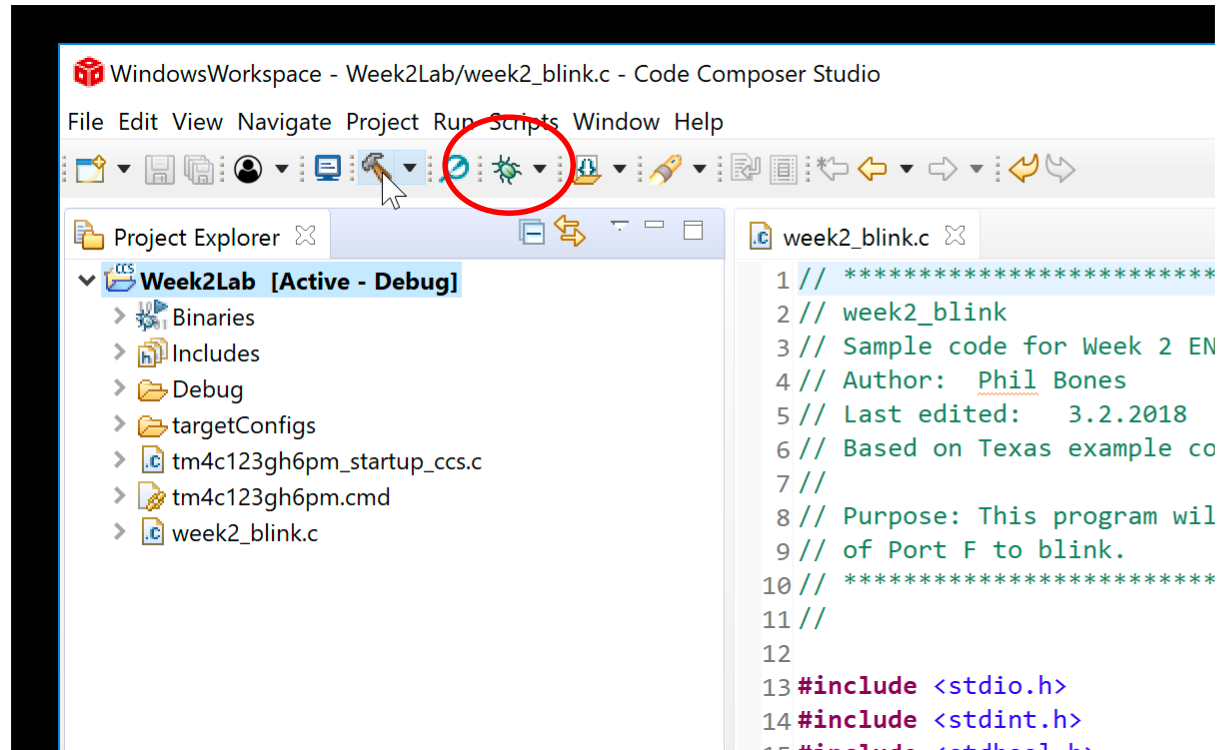
- Under the menu item “Project” select “**Build project**”. There is an icon that represents this in the form of a *Hammer*



- You should have a successful build, as shown in the console box.
- You’ve built (compiled and linked) your first project (.out file still on your P: drive)
- Let’s test it in debug mode

Loading and debugging your project

- After unwrapping your Tiva board and checking the contents, use the USB cord to connect your board to one of the two front USB sockets on your bench PC
 - The device driver should already find your board. If not, consult a TA
- Click on the **Bug icon**



- If you get asked for a "New Target Configuration", choose "Yes", "Stellaris In-Circuit Emulation" and "Tiva TM4C123GH6PM"
- The file **NewTargetConfiguration.ccxml** will appear in your project directory under **targetConfigs**

Debug

Week2Lab [Code Composer Studio - Device Debugging]

Stellaris In-Circuit Debug Interface/CORTEX_M4_0 (Suspended - HW Brea

main() at week2_blink.c:29 0x00000640

_c_int00() at boot.asm:227 0x000007CA (_c_int00 does not contain fra

Name	Type	Value	Location
(x)= clock_rate	unsigned int	0	0x20000200

```
53 // Write a zero to the output pin 3 on port F
54 GPIOPinWrite(GPIO_PORTF_BASE, GREEN_LED, 0x00);
55
56 // Enter a gadfly loop (kernel) to make the LED blink
57 while (1)
58 {
59     //
60     // Delay (passing the argument value clock_rate / 3 gives a delay of 1 sec)
61     //
62     SysCtlDelay(clock_rate / 12);
63
64     //
65     // Turn on the LED
66     //
67     GPIOPinWrite(GPIO_PORTF_BASE, GREEN_LED, GREEN_LED);
68
69     //
70     // Delay
71     //
72     SysCtlDelay(clock_rate / 12);
73
74     //
```

Console

Week2Lab

CORTEX_M4_0: GEL Output:
Memory Map Initialization Complete

Note: This message indicates that the loader has successfully installed the new program in the Tiva program memory.

Run and debug your program on your Tiva

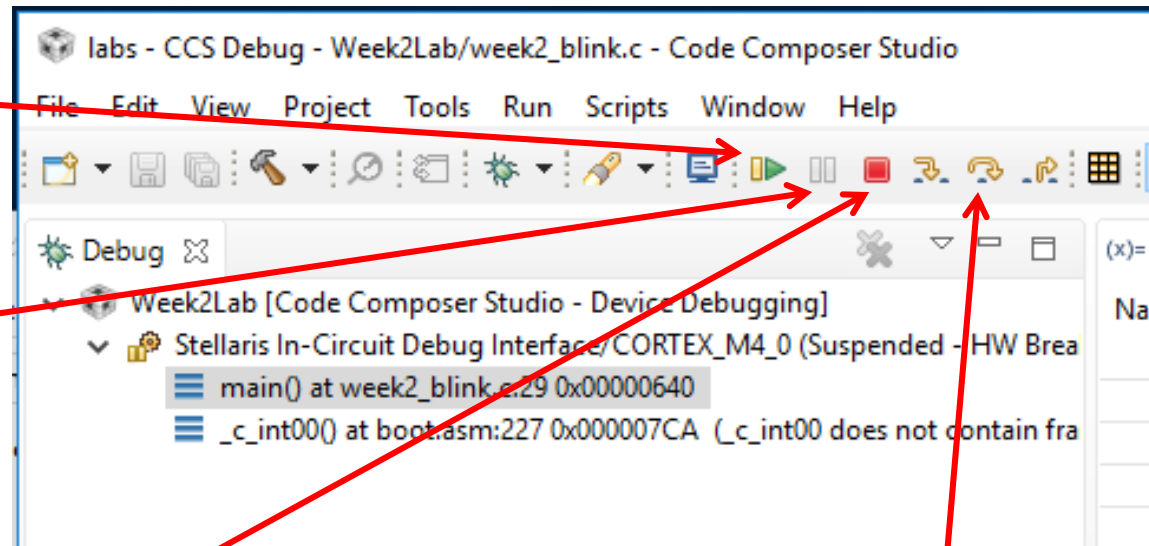
Also see:

http://processors.wiki.ti.com/index.php/GSG:CCSV5.0_Debugging_projects

- To run your program that you have already loaded into the Tiva memory, click-on the **green arrow**

- To stop the program, click on the two **yellow parallel lines**...

- Close the debugger by clicking on the **red square**... or go to the top right utility and switch between debug and edit mode (much faster!).



In the Help screen for CCS, select “stepping” in the index for more information.

- To single-step (“**step over**”), click on the right yellow arrow...

Using the debugger

- Click back on **week2_blink.c** and find TivaWare function, **GPIOPinWrite()**
- Set a **breakpoint** on the GPIOPinWrite function
 - Do this by clicking on the line associated with this function at the left of the window (in the blue hash region)
 - A **dark blue diamond** shape will be inserted in this blue region, associated with the function line number
- Resume your program execution but pushing F8 or clicking **the green resume arrow**
 - Did your program stop?
 - What is the status of the green LED?
 - What happens if you press F6 or the “Step over” button?

4.2 Compile, link and load OLEDTest.c

The build of this program needs access to the ustdlib module and to the OLED support module.

ustdlib.c: Right-click on the week3Lab project, **Add Files**

browse to **C:\ti\TivaWare_C_Series-**

2.1.4.178\utils\ustdlib.c Open, then choose **Link**

ustdlib.c will appear in the project with a link icon alongside.

OrbitOLED: **File | New | Folder | Advanced >> | Link to alternate location**

Browse to the OrbitOLED folder in your workspace, click **Finish**

The OrbitOLED folder will appear in the project with a link icon alongside.

Add **P:\Courses\ENCE361\labs** (i.e., your workspace containing the OrbitOLED folder) to the compiler include paths,

using **Properties | ARM Compiler | Include Options** and **+**