Guide to Using Texas Instruments Code Composer Suite version 7

Code Composer is a free IDE by Texas Instruments and this guide uses pictures taken from the computer screen as visual aids to help navigate through writing an Assembly program

(Compiled by James Kretzschmar, AE7AX)

The following will guide you there for the download. You will have to register with them before you will be able to complete the download.

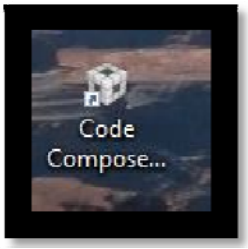
1. **www.TI.com**
2. Tools & Software
3. Software & Development Tools
4. Code Composer Studio - IDE
5. Visit the Code Composer Studio tool folder
6. Go to "Order Now" and click on "CCS-FREE" (red Download button)
7. Download the latest CCS

The Code Composer IDE is a large program and it will take a while to download and also you may need to turn off some of your security protection so it does not hit a snag.

Learning to program a microcontroller can be difficult. Even though there is a lot of material on the internet sometimes these references are not that useful and it easy to become frustrated. This guide will walk through step by step on how to program a simple assembly code program into a MSP430G2553 microcontroller using the LaunchPad Development Tool (MSP-EXP430G2).

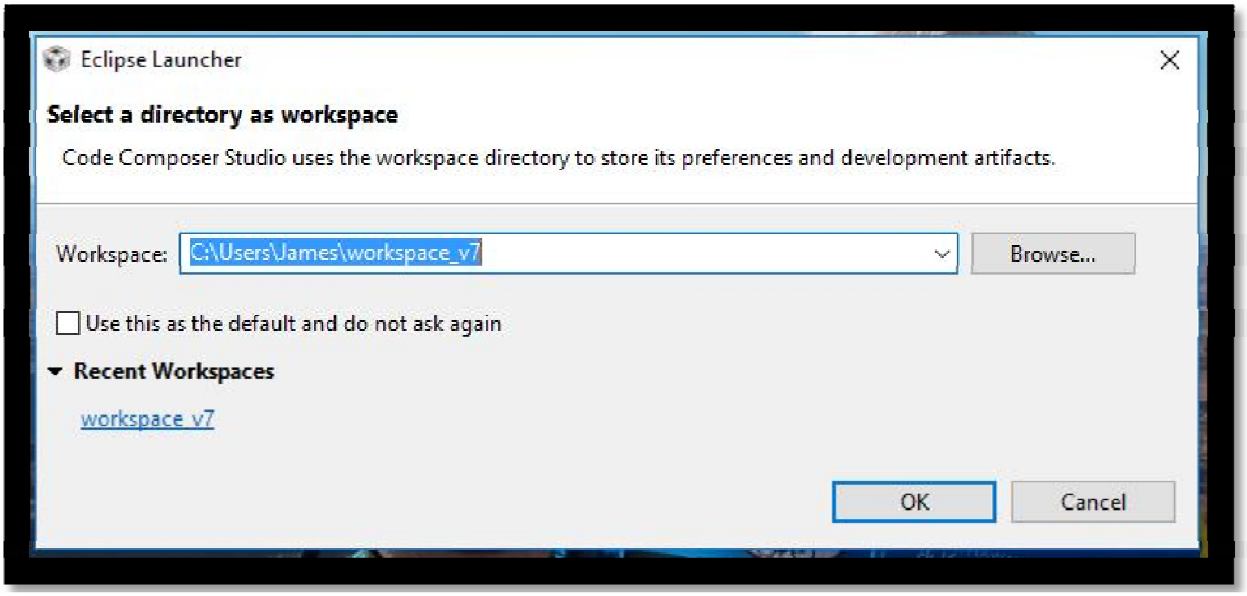
STEP #1

Once the Code Composer IDE is loaded on your computer the following icon should be on your desktop.



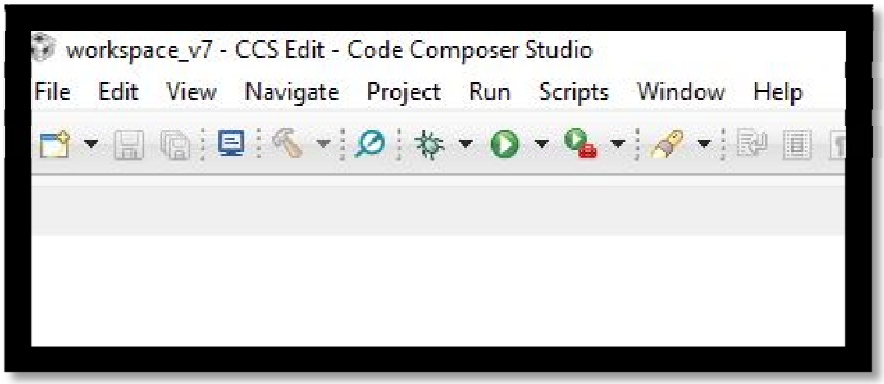
STEP #2

Upon entering into Code Composer the first screen you will see is telling you where your projects will be saved on your computer. This is where you will go to find projects that you have previously worked on, or where new projects will be stored. Hit "OK" and continue.



STEP #3

You will now see the screen below. Go to the "Project" button at the top, click on this button and a drop down menu will appear. Select "New CCS Project". If you have a saved program you want to work with, go to "File" then "Open" and find your project in the workspace_v7 folder. Click on the blue file folder icon that says "main".

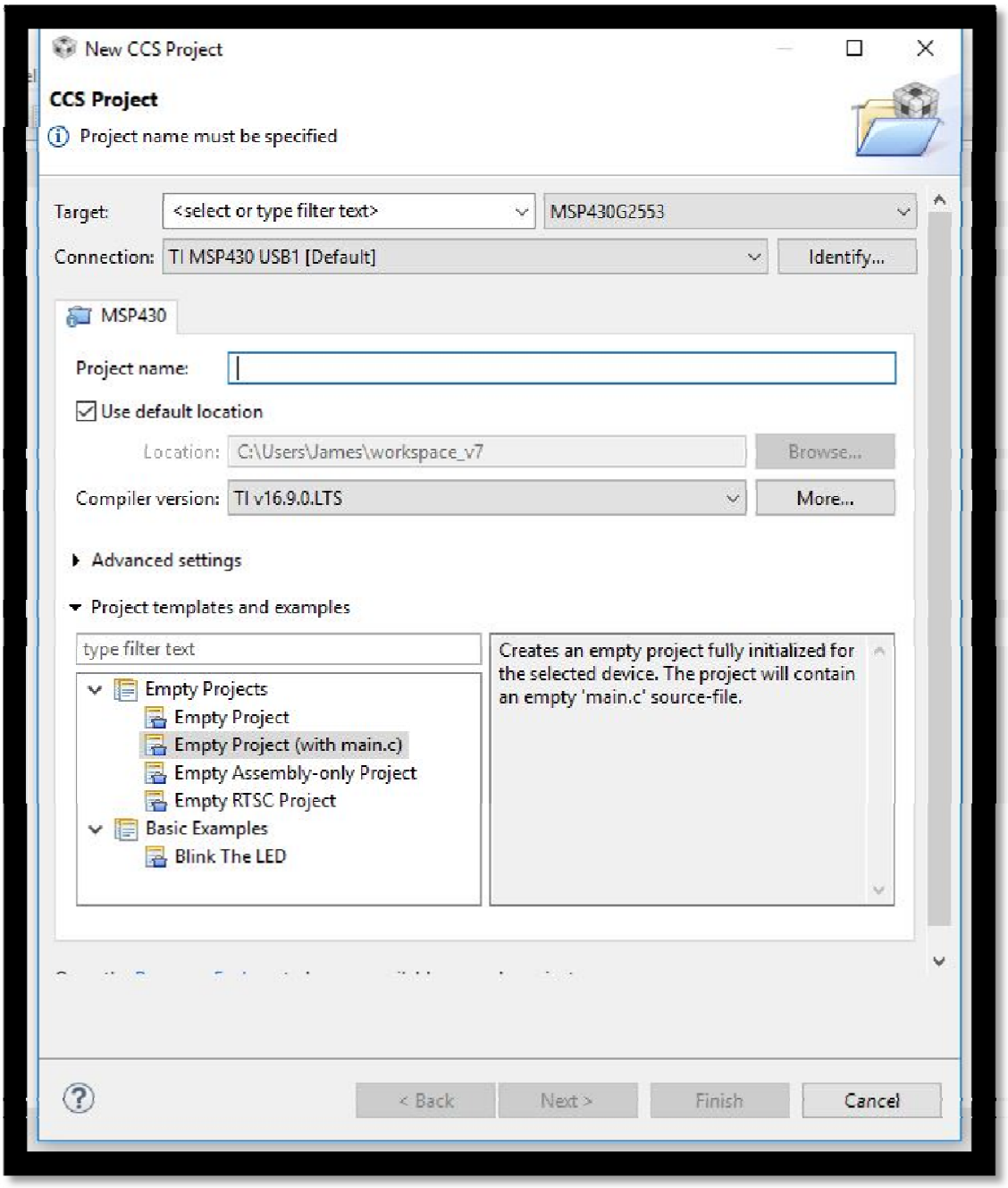


STEP #4

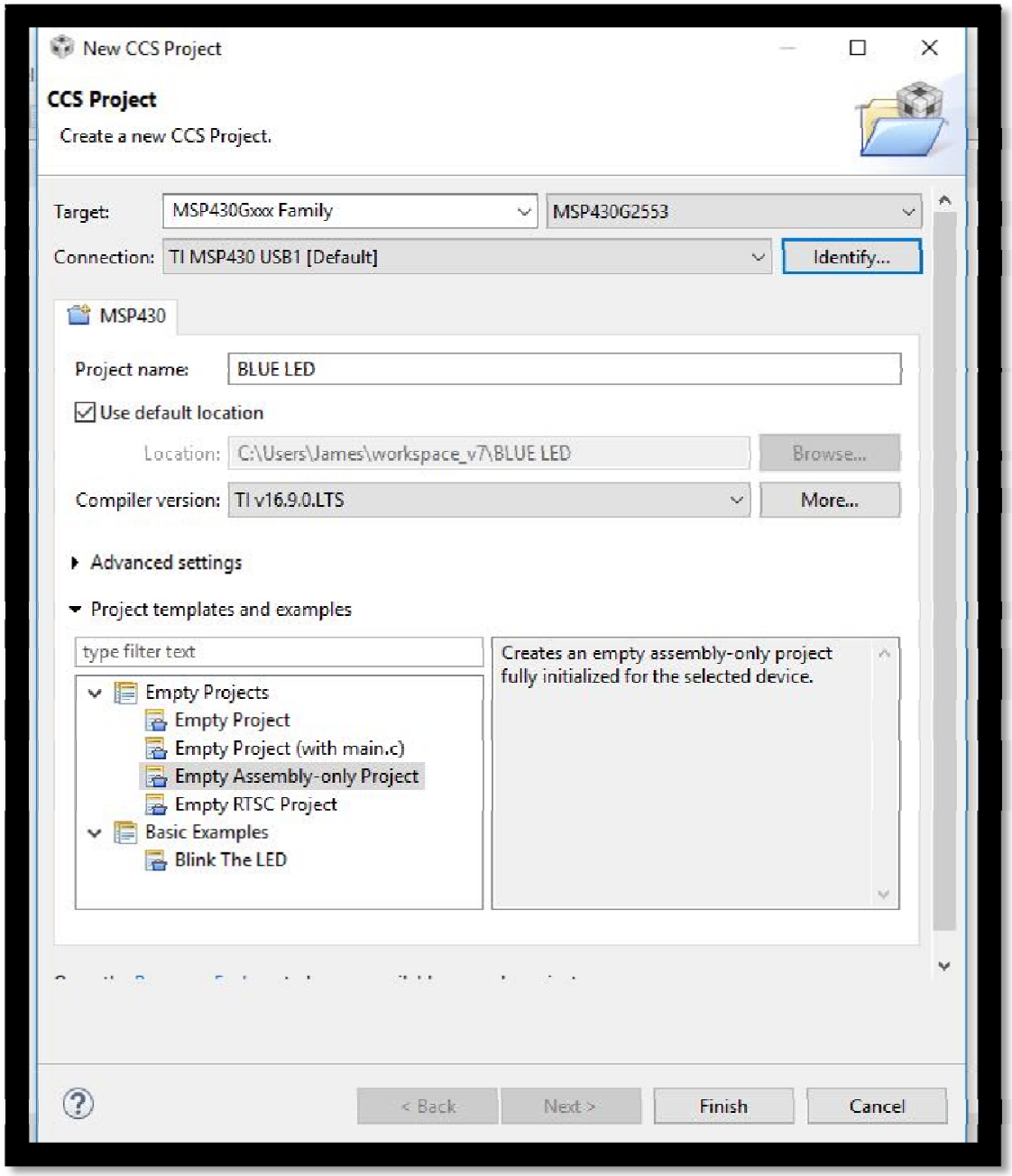
This next screen allows you to name your project for saving in the workspace, select the family of microcontroller you are using, select the microcontroller chip you are using, find the USB COM port on your computer that Code Composer wants to use (this will probably be a trial and error process), and select what type of empty project you want

(C or assembly). Selecting the "empty project type" gives you a framework to start with that has all of the setup information in place (ie. header files). All you need to do is then put in your code.

For "Target:" ... from the drop down menu select "MSP430Gxxx Family". From the next box to the right select the chip you are using "MSP430G2553". For "Connection:" click on "Identify" ... this will search for which USB COM port you have your board plugged in to. When you have found the correct USB port you will see a message box that says "Debug server session is running ... " ... click on the "Finish" button. For "Project Name:" ... give a name to your project. Next select what type of empty project you want to write code for. Once all the boxes have been completed you are allowed to proceed by clicking on the "Finish" button.



A completed example of the above screen.



STEP #5

The following framework for writing an assembly program appears. It is now up to you to enter your code. For example, in lines 21-22 you might define the clock speed for your microcontroller (ie. 1 MHz is default, 8 MHz, 12 MHz, or 16 MHz), or designate input/output pins. The details for setting up your particular parameters can be found in the datasheet for the microcontroller you are working with. For the MSP430G2553 microcontroller detailed information can be found in Texas Instruments document SLAU144J.

main.asm

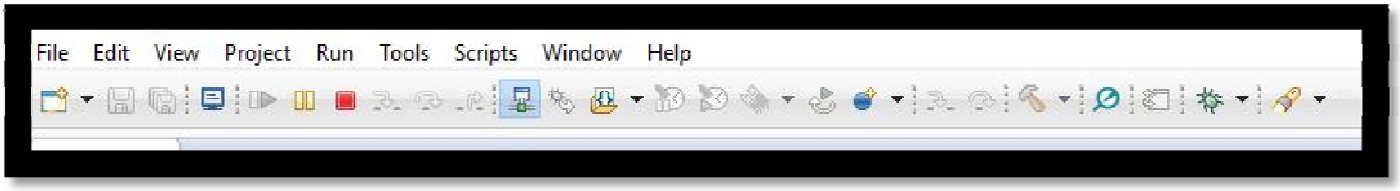
```
1  ;-------------------------------------------------------------------------------
2  ; MSP430 Assembler Code Template for use with TI Code Composer Studio
3  ;
4  ;
5  ;-------------------------------------------------------------------------------
6              .cdecls C,LIST,"msp430.h"       ; Include device header file
7
8  ;-------------------------------------------------------------------------------
9              .def    RESET                   ; Export program entry-point to
10                                             ; make it known to linker.
11 ;-------------------------------------------------------------------------------
12             .text                           ; Assemble into program memory.
13             .retain                         ; Override ELF conditional linking
14                                             ; and retain current section.
15             .retainrefs                     ; And retain any sections that have
16                                             ; references to current section.
17
18 ;-------------------------------------------------------------------------------
19 RESET       mov.w   #__STACK_END,SP         ; Initialize stackpointer
20 StopWDT     mov.w   #WDTPW|WDTHOLD,&WDTCTL  ; Stop watchdog timer
21
22
23 ;-------------------------------------------------------------------------------
24 ; Main loop here
25 ;-------------------------------------------------------------------------------
26
27
28
29 ;-------------------------------------------------------------------------------
30 ; Stack Pointer definition
31 ;-------------------------------------------------------------------------------
32             .global __STACK_END
33             .sect   .stack
34
35 ;-------------------------------------------------------------------------------
36 ; Interrupt Vectors
37 ;-------------------------------------------------------------------------------
38             .sect   ".reset"                ; MSP430 RESET Vector
```

Sample code for a simple oscillator:

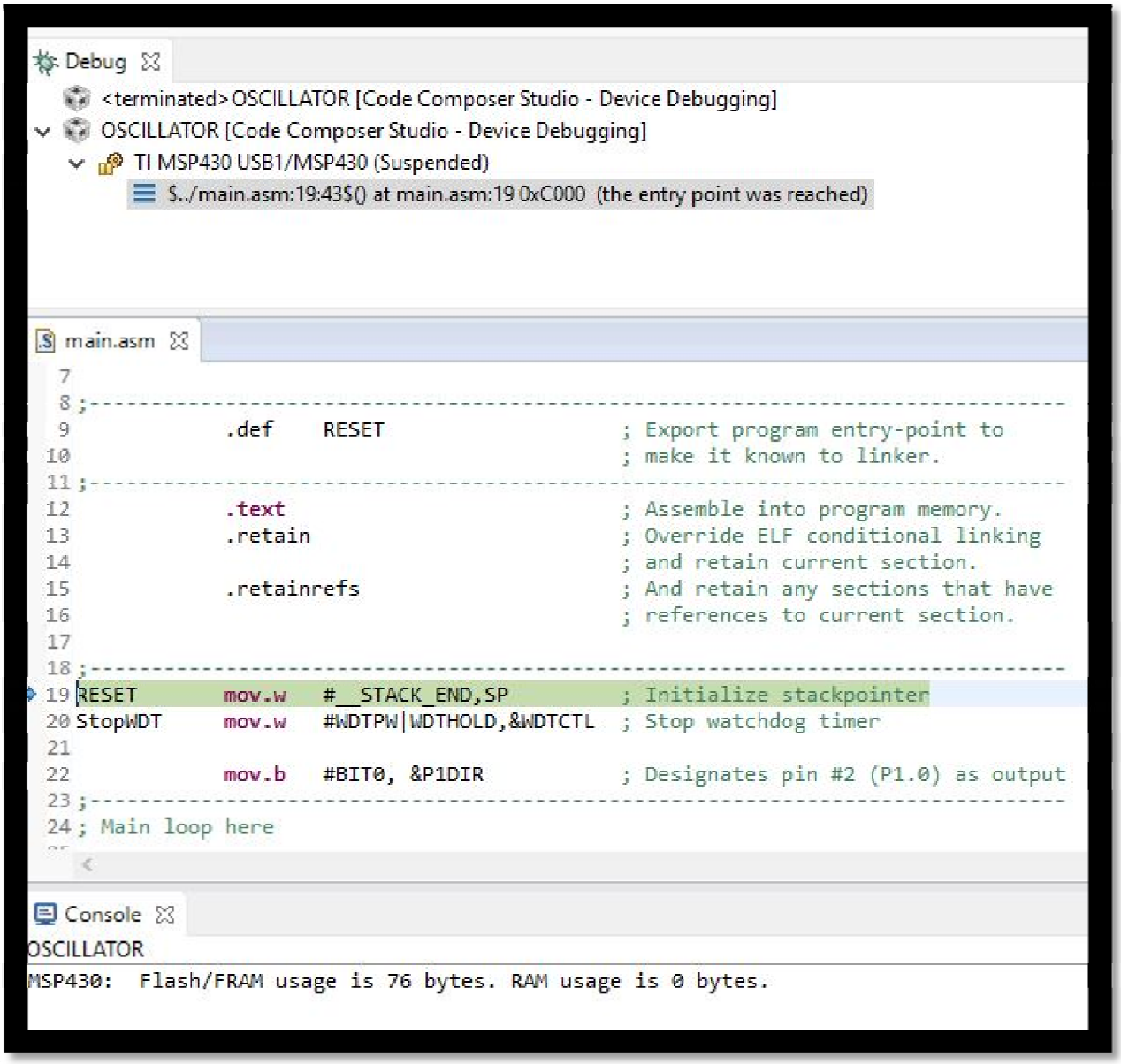
```
;-------------------------------------------------------------------------------
RESET       mov.w   #__STACK_END,SP         ; Initialize stackpointer
StopWDT     mov.w   #WDTPW|WDTHOLD,&WDTCTL  ; Stop watchdog timer

            mov.b   #BIT0, &P1DIR           ; Designates pin #2 (P1.0) as output
;-------------------------------------------------------------------------------
; Main loop here
;-------------------------------------------------------------------------------

OSCILLATE:                                  ; Pin P1.0 goes on/off at rate of 67.4 KHz
    bis.b   #BIT0,&P1OUT                    ; P1.0 goes "ON", + voltage at pin #2
    nop                                     ; delay, no operation
    nop                                     ; delay
    nop                                     ; delay
    bic.b   #BIT0,&P1OUT                    ; P1.0 goes "OFF", no voltage at pin #2
    nop                                     ; delay
    nop                                     ; delay
    nop                                     ; delay
    jmp     OSCILLATE                       ; Go to "OSCILLATE" (a loop)

;-------------------------------------------------------------------------------
; Stack Pointer definition
;-------------------------------------------------------------------------------
            .global __STACK_END
            .sect   .stack

;-------------------------------------------------------------------------------
; Interrupt Vectors
;-------------------------------------------------------------------------------
            .sect   ".reset"                ; MSP430 RESET Vector
            .short  RESET

```

STEP #6

When you have the code written, go to "File" and "Save" your work. Next click on the "BUG" button (looks like a green beetle). This will debug the program and show any errors you may have.

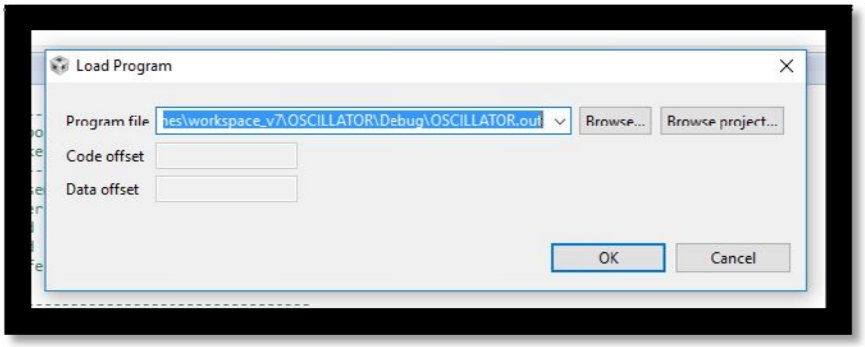


If there are no errors ... a "Build Finished" will briefly appear then you will see the following screen.



STEP #7

Go to the "Run" button at the top and drop down menu will appear. Click on "Load" and then "Load Program" when it appears to the right. The following screen will appear and click "OK".



Your program should now be loaded into the MSP430G2553 microcontroller and running. If your program was suspended for any reason (ie. making changes), you may have to go back into the "Run" drop down menu and select "Resume". Congratulations, now have fun developing all sorts of fun projects (Assembly code or C code) with the Texas Instruments MSP-EXP430G2 LaunchPad development board and MSP430G2553 microcontroller.

NOTE: I have found that programming in assembly code gives greater control for getting your microcontroller to do exactly what you want it to do. The higher level languages such as C, and the MSP430 specific ENERGIA (similar to Arduino coding) are nice and allow you to develop a project quickly, however you cannot really get "under the hood" and manipulate timing parameters, and other specifics easily.

==========