The University of Texas at Dallas
CE 4370 Embedded Microprocessor Systems
Laboratory 1: Introduction to Code Composer Studio

1 Laboratory objectives

- Familiarize with MSP-EXP432P43P401R development kit (TI Launchpad)
- Familiarize with the development software (Code Composer Studio)
- Create a project in Code Composer
- Add external resources to the project, compile it and execute it
- Use the debugger to insert breakpoints, watch registers and variables and control the execution flow.
- Familiarize yourself with Makefiles
- Release vs. Debug versions

2 Introduction

Code Composer Studio (CCS) is the integrated development environment for TI's DSPs, microcontrollers and application processors. It includes compilers for both of MSP432 low power 32-bit microcontroller device families, source code editor, project build environment, debugger, profiler, simulators and many other features.
This lab describes how to make a project and set the target configurations for - MSP432P401R

The MSP432 launchpad board is an evaluation board meant to assess the capabilities of any MSP432 device. The MSP432P401R has a 32-bit ARM cortex M4 MCU, 256 KB Flash and 64KB SRAM and an operating frequency of up to 48 MHz. 2 push buttons, 2 general purpose LEDs

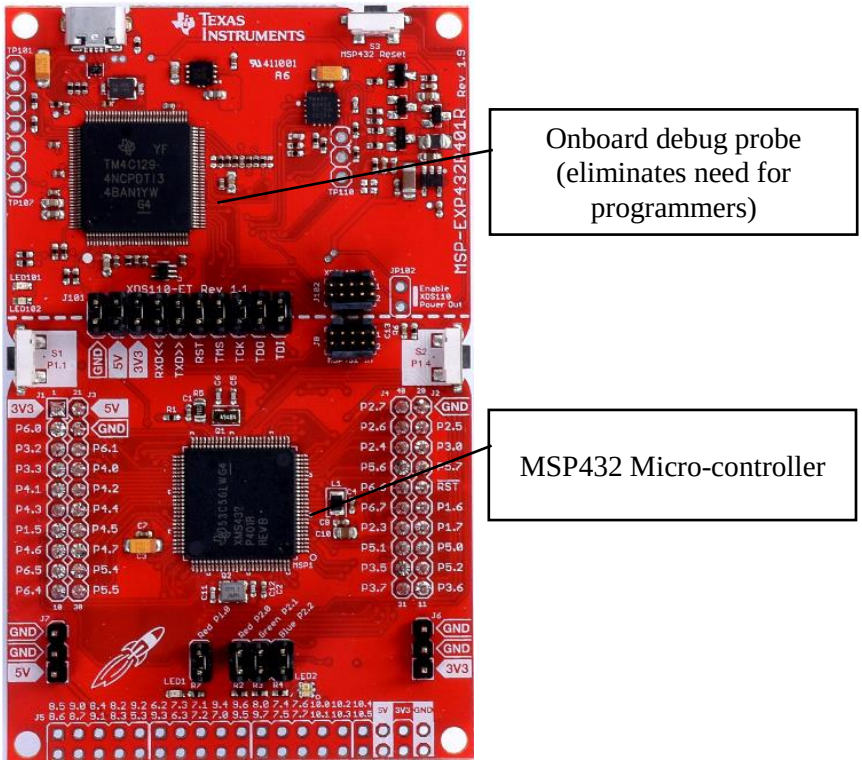


Fig.1 MSP-EXP432P401R Launchpad Development Kit

Pre-Lab
Read EXP432P401R Launchpad Development Kit user's guide manual and learn how to assemble and setup the boards for debugging

Tool Requirements
Software:
Code Composer Studio v8.x/9.x (This tutorial is based on CodeComposer v8/v9)
Hardware:
MSP-EXP432P401R Launchpad Development Kit

3 Lab Procedure
Compiler Installation - Perform if you are using your personal computer and compiler is not installed

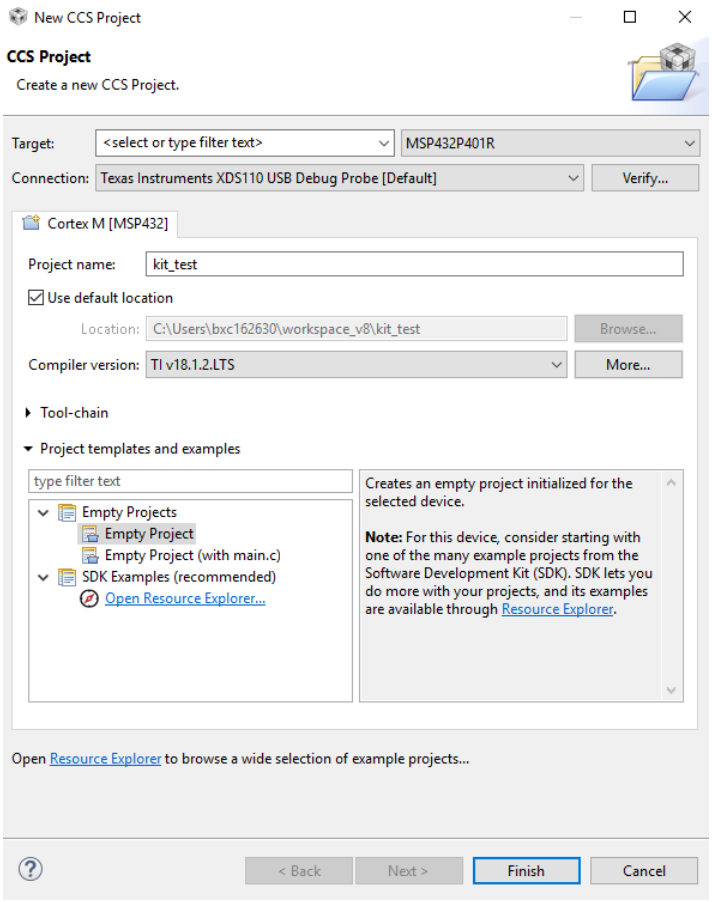
Code Composer Studio v8.x can be downloaded directly from TI website:
http://processors.wiki.ti.com/index.php/Download_CCS
and then follow the instructions to install and activate it:

Note: CCS can be used in software simulator mode without any external hardware and it is very useful for practicing and debugging your codes. For this, a MSP432 simulator is needed.
Some newer CCS versions do not come with a simulator anymore and thus, a hardware board is required to run and debug the code.

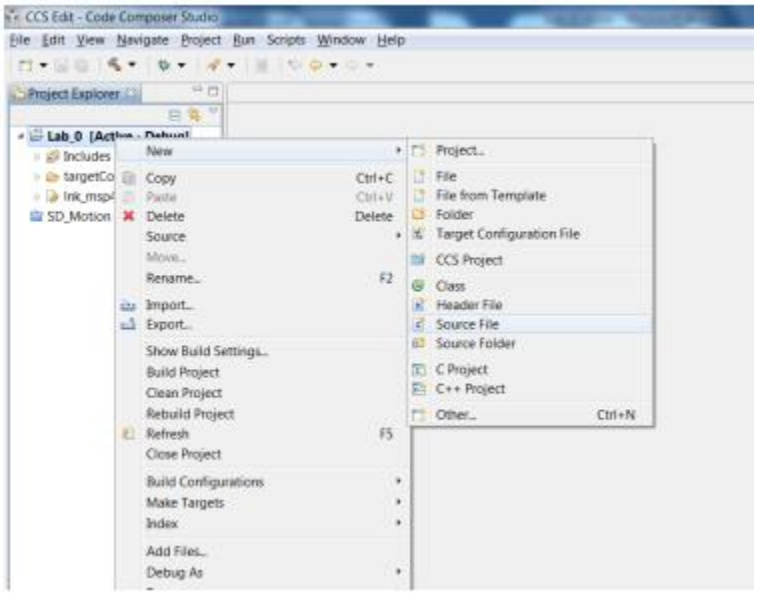
4.1 Create New Project for MSP432

Create a new project and make a target configuration file for MSP432

- Open CCSv8
- Click on *File* → *New* → *CCS Project*
- In the *Project name* field choose a name and a location for the project
- In the Device section select Family as MSP432 family
- Select MSP432 Family and MSP432P401R as Variant (same as on the Launchpad kit)
- Select connection type Texas Instruments XDS110 USB Debug Probe[Default]
- Select Compiler version : TIV18.1.2.LTS – or any other TI compiler version (not gnu)
- Select Empty Project and click Finish

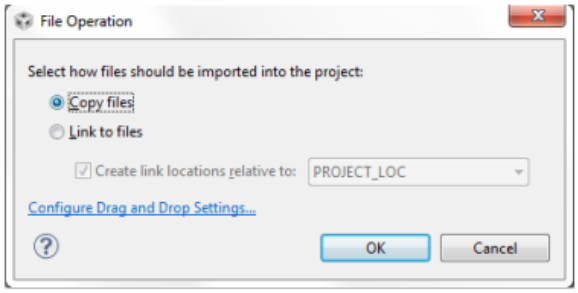


An empty project is created. You can create source or header files and add them to the project by selecting *Project Explorer* the left side of main view and then right-click and select *New*.

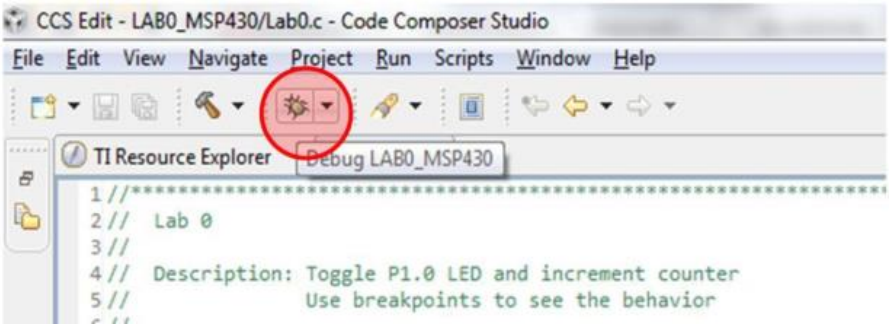


In order to add existing file to the project in *Project Explorer* right-click and select *Add Files*… or go to *Project* →*Add Files*

Select <Lab1_LEDs.c> which is available on the course elearning page. You can either copy the file into project folder or link to it in the import method dialog box.
Make sure that the extension of the file is .c if you cut and paste it into a new file.



Now you will compile and debug the project with F11 or by clicking on the green bug icon. Make sure the explorer board is connected to the PC (power LED is on) and the driver for USB device is installed and power sel switch set to ez (not JTAG).



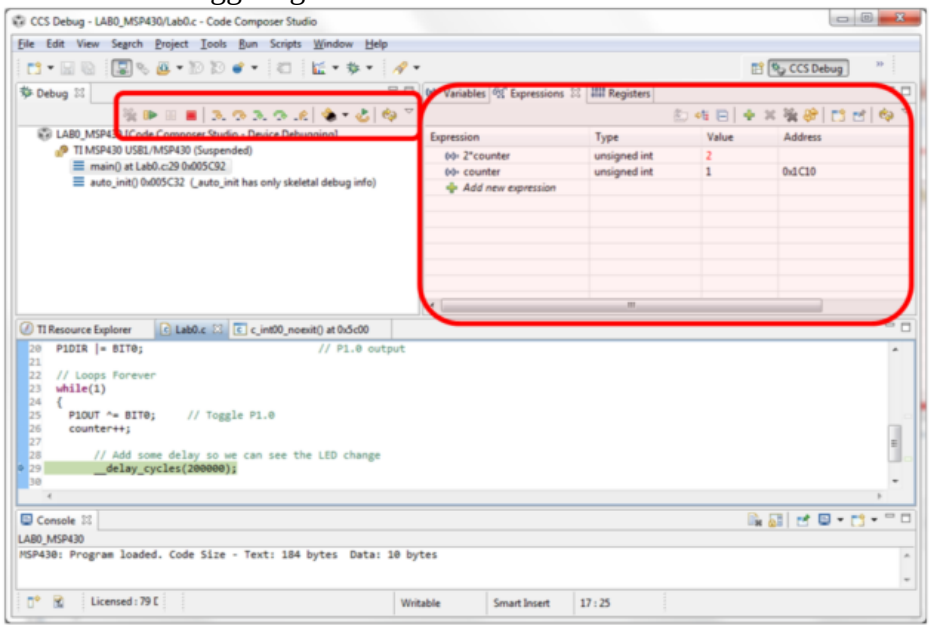
If the hardware setup is correct and no error exists in the code (ignore compiler warnings), debugger will start and the screen changes to the debug perspective. Press F8 to resume the code execution. LED will start blinking.

Show the working design to any TA/lecturer and get your lab sheet signed.

Follow the instruction in the lab sheet and change and record different delays by changing the delay loop. Answer the questions in the lab sheet.

Part-1 Debugging

Launch the debugger again. The screen below will be shown:



In the source code view, the first line to be executed is highlighted. Here, it is the first line after *main*. To control the debugger, use debugging toolbar on the top-left side of the screen. Debugging functionalities are Run, Halt, Terminate, Step into/over/return. Become familiar with their functionalities. You can watch Registers and Expressions in the watch list. It is also possible to see the values by placing the mouse curser over variables in the source code view.

Answer the questions in the lab sheet.

Part-3 Modifying Makefile

Open a DOS terminal window on your computer (type cmd in search window). Go to your project folder using 'cd' command and look at the folder contents using 'dir' command.

Locate the folder with the Makefile and type:

C:\Your path to lab0 >C:\ti\ccsv8\utils\bin\gmake make -k all

The gmake utility reads the Makefile and follows the compilation instructions there to compile and link the source code in order to get the executable binary.
Open the Makefile and locate the name of the binary file. Change it to any other utdallas.out. Change also:

```
EXE_OUTPUTS += \
Lab1_LEDs.out \
EXE_OUTPUTS__QUOTED += \
"lab1_LEDs.out" \
BIN_OUTPUTS += \
Lab1_LEDs.hex \
```

BIN_OUTPUTS__QUOTED += \
"lab1_LEDs.hex" \

to utdallas as well as the any reference to lab1 at the tools invocation section.
Clean the previously generated files by typing:

C:\Your path to lab1 >C:\ti\ccsv8\utils\bin\gmake clean

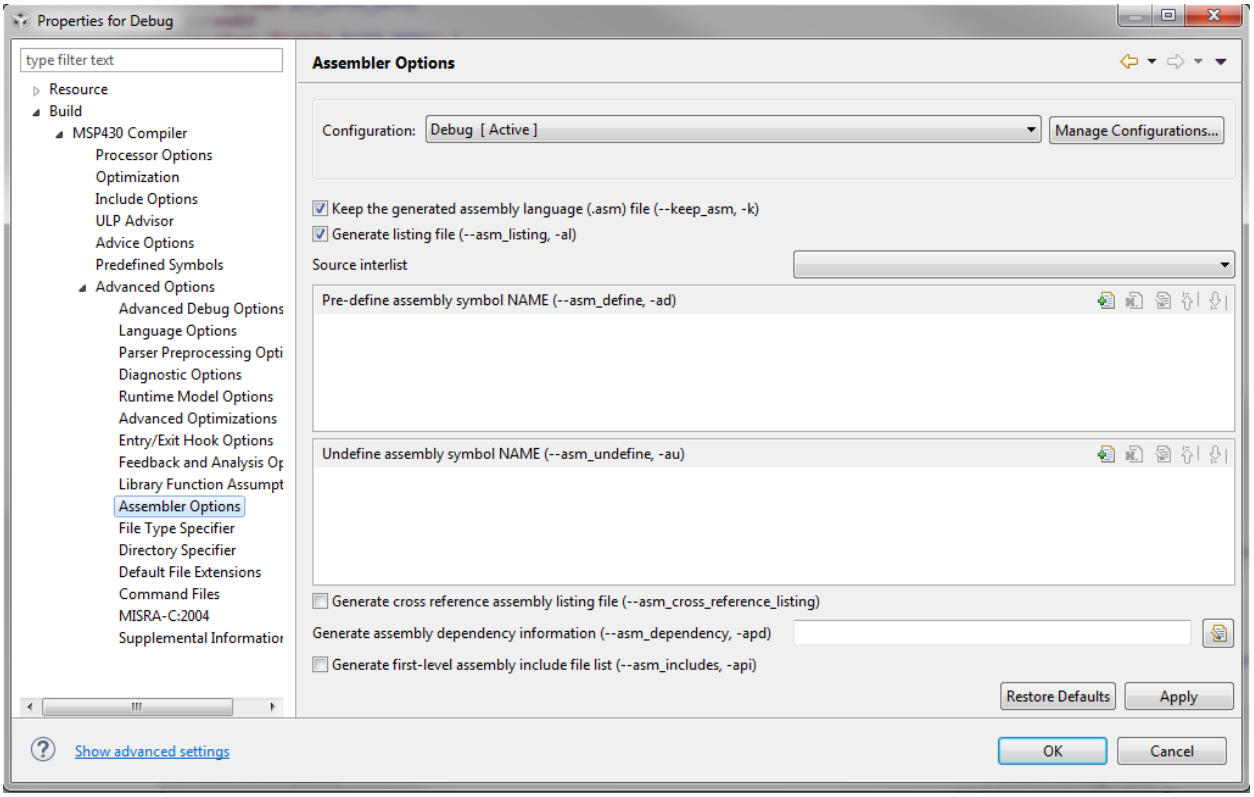
This will read the Makefile and follow the instructions in the clean section (typically deleting previously generated output files)

Then recompile the file. This should generate a new binary file with the new name.

Answer the questions in the lab sheet

Part-4 Assembly

- Right click on Debug folder → Properties → Build → Advanced options → Assembler options. Select "Keep the generated assembly language" and "Generate listing file" as shown below.
- Apply changes
- Build clean
- Build all

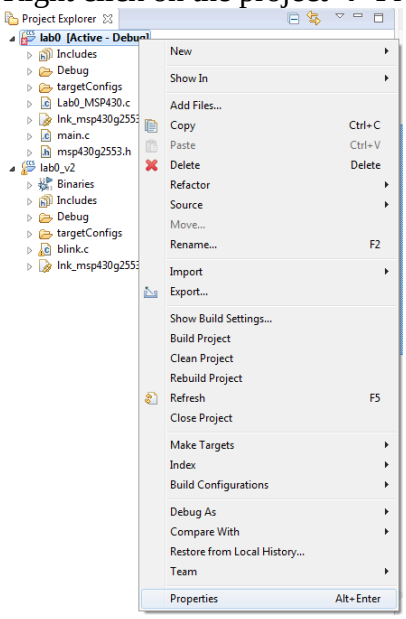


If you open the Makefile you will see that at the tool invocation line the options -k --asm_listing have been added.

A lab1_LEDs.asm file should have been generated. Open the file and locate the main execution loop and answer the question in the lab report sheet.

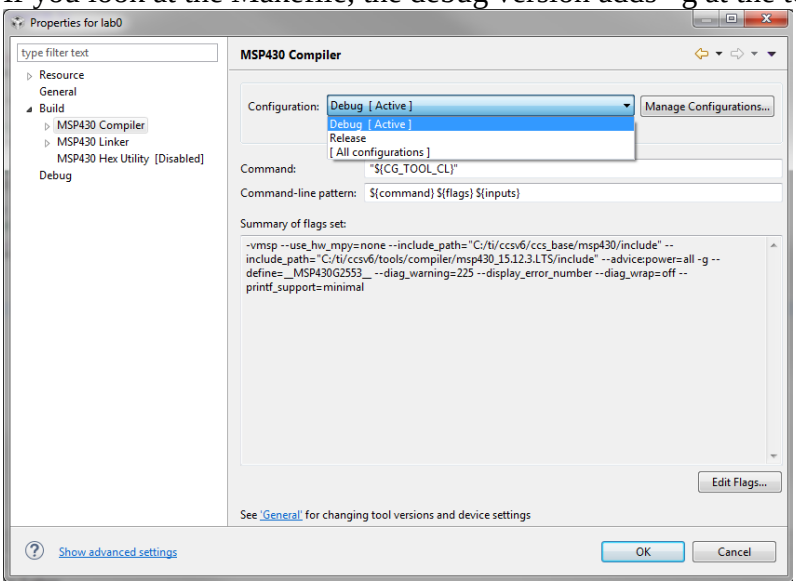
Part-5 Configuration

Right click on the project → Properties



Recompile the code changing from Debug to Release. Click on Manage Configurations button and select the active configuration.

If you look at the Makefile, the debug version adds –g at the tool invocation line.



Record the size of the binary (executable .out) file generated in both cases. Annotate in the lab report sheet.