ECE315 Introductory Microprocessor Laboratory

Lab 6

Embedded Software

1. Introduction

In Labs 1-5, you designed and assembled a fully custom PCB. Now you load some basic test software onto your board to verify that it has been designed correctly.

2. Creating Default Software Project

ECE315 utilizes a custom embedded learning platform that uses an Infineon PSoC4 microcontroller (CY8C4045AZI-S413). You will use VS Code for programming and debugging PSoC4 based devices. This document will instruct you how to set up a base project for the PSoC4 used in class.

3. Computer Aided Engineering

The College of Engineering supports various computer labs on the engineering campus. These labs are supported by a group called Computer Aided Engineering or CAE. CAE supplies Windows based PCs throughout the engineering campus that can be accessed by any engineering student. In order to get access to these PCs, you need to create a CAE account. These PCs have all the required programs and drivers needed for the software we will use in ECE315.

4. Installing Software on a Personal Laptop

Your second option would be to install the required software on your own laptop. All of the software on your personal computer. All of the software used in the course runs on Windows, Mac, or Linux operating systems.

If I had to choose which of the options to use for ECE315, I would probably recommend using your own laptop. There are some minor, yet annoying, situations that arise from using the CAE based PCs in conjunction with ModusToolBox. Workarounds for these annoyances will be handled as part of the in class exercises.

If you do choose to use your own laptop/PC, know that ECE315 staff are not IT professionals. We can give you general advice, but if the software on your laptop is not working properly, then it is your responsibility to make sure it gets fixed. Turning in late assignments due to technical issues with student owned equipment will not be accepted as a reason for work being turned in late.

You will need to download and install the following programs. Follow the default installation instructions for each piece of software. When choosing installation

directories, you should never include spaces in your directory names. Spaces in file/directory names cause all kinds of problems.

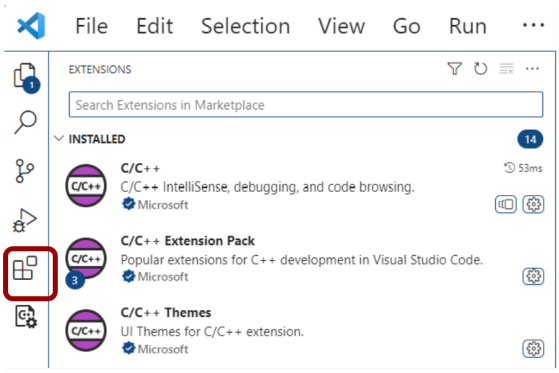
- ModusToolbox v2.4.1 (Do Not Install v3.x.x)
- ModusToolBox Patch File
- VS Code

5. Installing VSCode Extensions

These steps are only required if you are installing VS Code on your personal computer.

a. Open VS Code

b. Click on the Extensions Icon



c. From the Search Bar, Install the following Extensions

i. C/C++ IntelliSense, debugging, and code browsing
ii. ARM Cortex-M GDB Debugger support for VSCode
iii. Doxygen Documentation Generation
iv. RTOS Views
v. GitLens (Optional)

d. Close VS Code

6. Source Code Directory

a. From modus-shell create a directory on the **C drive** using the following command. If you are using a CAE PC, be sure to use your CAE username in place of "username". If you are using your personal computer, then use the username you use to log into that PC. **DO NOT create a directory with any spaces in its name!**

If you are using the CAE PCs, you will still want to use the C: drive. The I: drive is very, very slow!

```
mkdir /cygdrive/c/Users/username/ece315
```

b. Enter the directory you just created

```
cd /cygdrive/c/Users/username/ece315
```

7. Copy the Starter Project from GitLabs

a. In modus shell, clone the starter repository from GitLabs

git clone https://git.doit.wisc.edu/engr/ece/cmpe_courses/public/ece315/mtb/f23/starter-project.git ./ece315-mtb-f23-starter-project

b. Enter the starter-project directory

```
cd ./ece315-mtb-f23-starter-project
```

c. Set up the project

```
make getlibs
make vscode
```

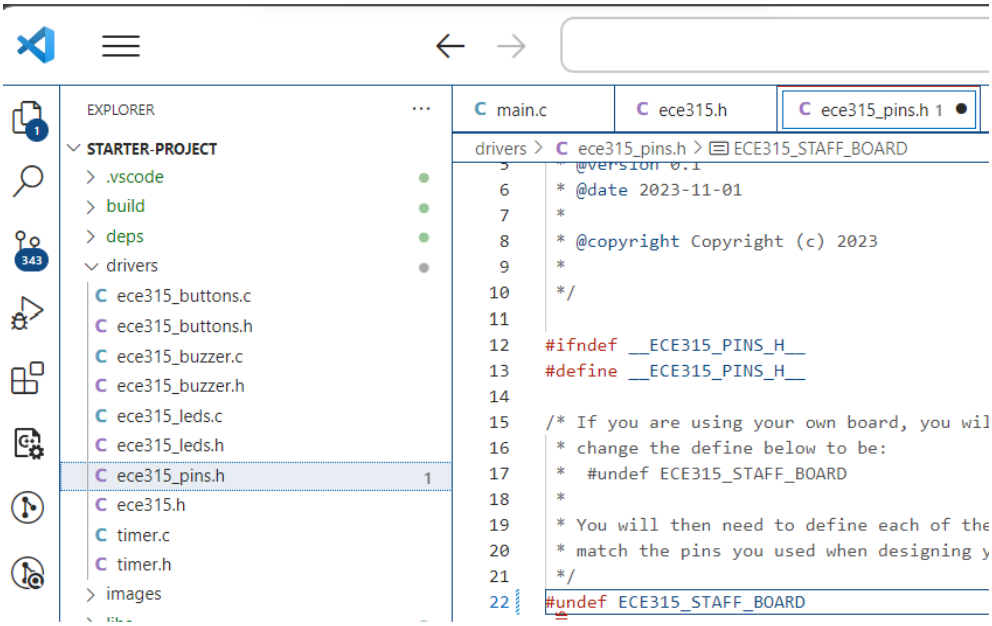
8. Program Your Board

a. In modus shell, start VS Code from the directory that contains your project

```
code .
```

b. In Explorer pane, expand the drivers directory and double click on ece315_pins.h
c. In ece315_pins.h, modify line 22 so that you undefine ECE315_STAFF_BOARD

```
#undef ECE315_STAFF_BOARD
```



d. Starting on line 65, modify the #defines to match the pins you choose to use in your design. You will need to look at your schematics to complete this part of the test software.

e. Build the project from the modus shell.

```
make build -j16
```

f. Place your programming/debugging header on your board. Be sure to align the VTARG and SWDIO pins correctly before plugging in the connector.

g. Connect the programmer with a microUSB cable to your PC

h. Program the board from modus shell

```
make program
```

i. The LEDs on your board should all be blinking On/Off. You can toggle the buzzer on/off by pressing the pushbutton.

   If the LEDs are not blinking ON/OFF or the buzzer does not work, verify that you have defined the macros in ece315_pins.h to match your schematics. If there are no errors, ask your TA for advice on how to debug your board. They can show you how to use the oscilloscopes in the lab to determine why an LED may not be turning on.

j. If you are using the CAE PCs, you will want to copy your project directory from the C: drive to the I: drive. CAE purges the C: drive every 48 hours.

```
cp -R /cygdrive/c/Users/username/ece315 /cygdrive/i/ece315
```