Department of Electrical and Computer Engineering

EEC 172 Spring 2015

Lab verification due at the end of your lab session Wednesday April 8 or Thursday April 9, depending on you lab section.

-5 points for verification on April 10. -10 point for verification during the week of April 13. Zero credit after April 17.

Lab report due during lab section April 15 or 16.

LAB 1: Development Tools Installation, Sensor readings, OLED display.

Objective:
This lab will cover the installation procedure for the Keil MDK, familiarize you with the Redpine Singals WYZBEE development board, and the Adafruit OLED Breakout Board. You are encouraged to install the tools we will use this quarter on your own laptop so that you can work outside of the lab.

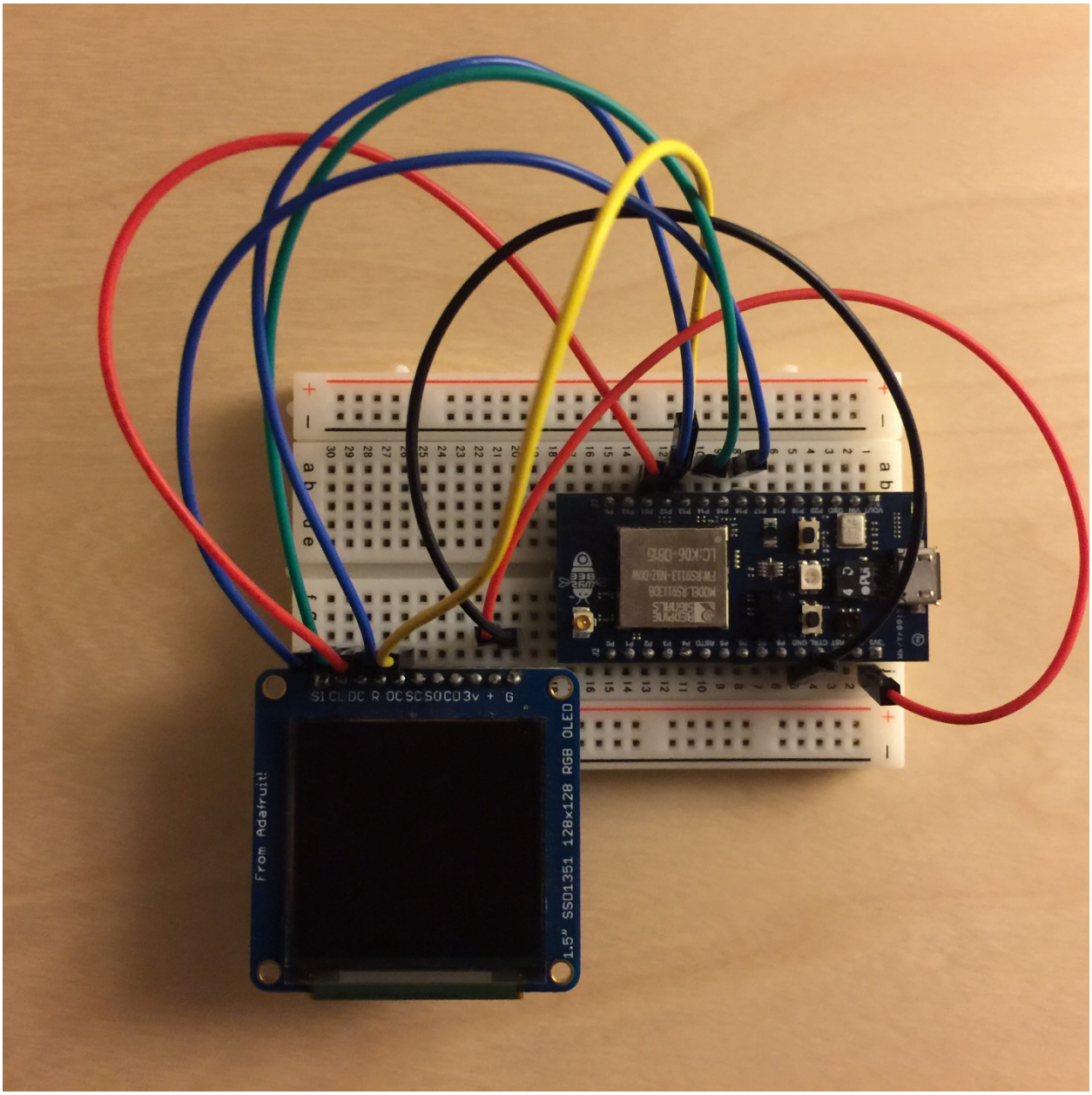
Tool installation:
If you wish to work from your personal computer, please follow the instructions below. If you intend to use the schools lab computers for your work, you may skip this step, as the tool are already installed. Follow the installation instructions for Keil found in the Wyzbee user guide: WyzBee_UserManual_V1.0.pdf to install the tools.

Without a software license, your code size is limited to 32K. The university owns licenses you can use to build larger executables. To access the Keil license server from outside Kemper Hall, you will need to use the COE VPN.

- Go to: https://vpn.engr.ucdavis.edu
- login with your kerberosID/kerberosPASSPHRASE
- Download the GlobalProtect software for your OS.
- Install GlobalProtect
- Configure Global Protect to connect to portal: vpn.engr.ucdavis.edu
- When you are connected to the COE VPN via Global Protect, you can then access the COE license server for KEIL.
- KEIL license server information:
- license.engr.ucdavis.edu port 8224

Board Wiring:
You will need to wire the OLED for use will the WYZBEE board. Follow the wiring in the table below:

UCDAVIS
COLLEGE OF ENGINEERING



WyzBee pin name	OLED Board pin name
GND	G

3v3	+
P16	SI
P14	CL
P11	DC
P12	R
P13	OC

The WYZBEE board by Redpine Signal is a brand new development board for prototyping Internet of Things applications. We are the first to use this board, with access before the world wide launch this quarter. Some of the features of the board include:

- A Spansion FM4 series Microcontroller unit (MB9BF568N). See http://www.spansion.com/downloads/Spansion-FM-ARM-Cortex-Microcontrollers.pdf for more details.
- An IR receiver.
- A proximity and ambient light sensor.
- A 9-Axis Motion and orientation sensor (accelerometer, gyroscope, and magnetometer) See http://www.invensense.com/mems/gyro/nineaxis.html for more details.
- A Wi-Fi and BlueTooth connection module.
- On board debug adapter.
- USB connection module.
- You will also us the ADAFRUIT 1.5" Color OLED Breakout Board. See https://learn.adafruit.com/downloads/pdf/adafruit-1-5-color-oled-breakout-board.pdf for more details.

Lab Tasks:
For this lab you should do the following and have your TA sign off on your work:

1. Create a simple program that changes the LED color on the Wyzbee board when the push-button is depressed.
2. Print the names of your lab group to the OLED.
3. Read the temperature sensor once and print the current temperature to the OLED.
4. Create a loop that reads the values of the Accelerometer, and Magnetometer and prints them to the OLED continuously.

Part 1.
SmartSite contains a zipped sample project called TriLED_APP.zip that cycles the color displayed by the tri-color LED. Download, compile and run the project. Read the code in main.c and understand how it works. Modify the project to change colors when the push button is depressed rather than change periodically.

Demonstrate your working program to your TA and have him or her sign your verification sheet

Part 2.
Invoke the Keil uVision tool and begin a new project. Configure you project as shown in WyzBee_UserManual_V1.0.pdf. This will configure your project for use with the Wyzbee board.

Redpine signals and AdaFruit industries supply APIs for interacting with the devices on the board which you can use to simplify your assigned tasks.

There is a shell project on Smartsite that you can use to build your program. It is setup to include all the files and libraries you'll need to complete this lab.

The OLED documentation contains information on how to initialize and use the OLED. Your code will look something like this:

```
Adafruit_SSD1351 myOled = Adafruit_SSD1351();//Constructor
myOled.begin();              //@ OLED screen Initialization
myOled.fillScreen(BLACK); //@ fills the OLED screen with black pixels
myOled.setCursor(0, 0);   //@ Set the print cursor to the initial location
myOled.setTextSize(3);    //@ Set the text size on the OLED
myOled.write("H");        //@ Write a character to the screen
```

You should write a function to prints a strings of a given color at a given location. You will use this function throughout the remainder of you labs. Take some time to explore the API to see what else is available.

Demonstrate your working program to your TA and have him or her sign your verification sheet

Part 3.
Read the Redpine signals API section regarding the use of the I2C. You will need to initialize the I2C module with a configuration structure. You can use this for now:

```
WyzbeeI2c_Config_t config ={
            100000,
            I2cMasterMode,
            I2cNoizeFilterLess100M,
            0x00,
};
```

You can now use the `WyzbeeI2c_Write` and `WyzbeeI2c_Read` functions to access the onboard sensors.

Information on the Temperature and Humidity sensor can be found at http://www.silabs.com/Support%20Documents/TechnicalDocs/Si7021.pdf. You'll need the data sheet to find:

a) The i2c Address of the sensor.

b) The address in the device containing the information you need.
c) How convert the raw sensor data in a relative humidity and temperature.

The information read from the sensors sometimes spans more than one address. For example, the temperature value returned by the sensor is a 16 bit value. You will read them the value over the i2c bus however as two consecutive bytes. You will then need to combine these two bytes to construct the actual temperature value. This can be achieved by shifting the most significant byte 8 bits to the left and adding the least significant bits to the value as shown below:

```
Int8_t msb, lsb;
//code to get the bytes goes here
int16_t temperature = ((int16_t)msb<<8)|lsb;
```

Note that the msb must first be cast as a 16 bit integer before you shift left (this is done by placing `(int16_t)` before the variable name). This lets the compiler know that although msb is only an 8 bit integer, it should be treated like a 16 bit number for this operation. You can read more about type conversion on Wikipedia: http://en.wikipedia.org/wiki/Type_conversion

Print the information in a digestible format to the OLED

Demonstrate your working program to your TA and have him or her sign your verification sheet

Part 4.
Create an application to continuously read the acceleration data from the 9-axis sensor. You can use the function `initMPU9250()` to initialize the accelerometer. Again, will need to use the data sheet to find:

a) The i2c Address of the sensor.
b) The address within the device containing the information you need.
c) How convert the raw sensor data into acceleration in terms of meters/second.

When you alter the orientation of the board, what happens to the acceleration readings? What is going on here?

Demonstrate your working program to your TA and have him or her sign your verification sheet

Lab Report (Due one week after Verification due date)

Your lab report should include:

- A brief description of how your program works
- A description of any noteworthy difficulties you encountered during your efforts
- A description of anything you think is interesting, unique or clever about your solution that might qualify for bonus points.

- Complete, well-commented source code for your program that is uploaded to SmartSite
- Your lab Verification Sheet.