ECE 2036: Lab #2 – *mbed* Hardware Starter Lab

(100 points)

Due Date: Monday Feb 17, 2020

Directions: To complete this lab, you will need to demo your working *mbed* projects to the TAs during their office hours. Once you get all your signoffs from the TA, please take a picture (or scan this page) and upload the image (or pdf) to canvas on Lab #2.

Student Name: _____ Final Turn-In Date: _____

Item	TA Signoff
Part 1. (10%) "What's up mbed!"	
Part 2. (20%) "Bouncing Ball"	
Part 3. (20%) "Digital Thermometer"	
Part 4. (20%) "Jazzy Tunes"	
Part 5. (10%) "Hello Micro SD Card"	
Part 6. (20%) "Triple Axis Accelerometer"	
All Checkoffs Completed on or before Feb 12 (+3% extra)	
All Checkoffs Completed on Feb 13 (+2% extra)	
All Checkoffs Completed on Feb 14 (+1% extra)	
Late Checkoffs 20% off per day (Sat &Sun count as one day)	

IMPORTANT: AFTER COMPLETING THIS LAB DO NOT TAKE APART YOUR HARDWARE BECAUSE YOU WILL USE IT FOR SUBSEQUENT MBED LABS IN THIS CLASS!!

Embedded devices account for 98% of the world's microprocessors. For every desktop computer, there are over 100 embedded devices. A high-end car can contain up to 100 microprocessors. Embedded devices contain a processor with software that is typically not changed by the user (called "firmware"). Most users are not aware that their cell phones, cameras, audio players, and TVs contain a computer with firmware. C/C++ is currently the most widely used language for embedded devices. ARM processors, which are similar to the one found in the *mbed* module, are used in about 80% of embedded devices including most cell phones.

This assignment will start with getting a variety of external components working with your *mbed* board. In addition to your *mbed* module, you will need a protoboard (a.k.a breadboard), wire kit, and LCD board which should come with your *mbed* kit that you purchased.

Instructions for Part 1 (10%): "What's up mbed!"

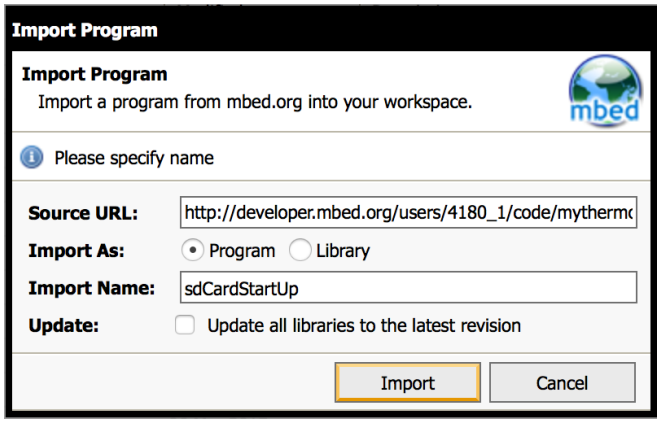
1. Read this link first. This tells you how NOT to break the pins on your *mbed* board! VERY IMPORTANT - COULD WIND UP COSTING YOU $50! This also gives you a brief tutorial on breadboards.
https://developer.mbed.org/handbook/Breadboard

2. Setup your *mbed* account. There should be instructions in your kit about setting up your account. You can look at the following youTube video to help you set up your account. Just choose the microcontroller platform of *mbed LPC1768* instead of the one listed in the video.
https://www.youtube.com/watch?v=BAzKg3vcB88

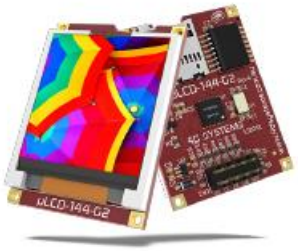
3. After placing the *mbed* board on the protoboard, you can connect it to the USB port on your computer. It should show up on your computer as an external drive. You will put files in this drive that will be automatically downloaded to the flash drives on your *mbed* system.

4. Import the skeleton code into your *mbed* account that is found at
http://mbed.org/users/4180_1/code/mythermostat/

You should import skeleton code to your *mbed* cloud account so that you can use the libraries provided for this and other labs. It should be very straightforward to click the "Import into Compiler" button. I would suggest populating the import screen in the following way. For the "Import Name" you can choose any name that you would like:



5. On your protoboard, please put the LCD on the side of the *mbed* chip where you have the pins 27-29. See the wiring table below for the pin connections needed. You might also want to read the *mbed* wiki page is (Please note some of the pin changes in the table in Figure 1!)
https://mbed.org/users/4180_1/notebook/ulcd-144-g2-128-by-128-color-lcd/



CAUTION

- If you switch the Rx/Tx pins, you will get the LCD splash screen message.
- MAKE SURE NOT TO SWITCH POWER AND GROUND PINS. If you do, you will have to BUY a new device!!!!

LCD Wiring for LCD test program

Mbed	uLCD pin
5V=VU	5V (1)
Gnd	Gnd (7)
TX=P28	RX (5)
RX=P27	TX (3)
P29	Reset (9)
Note: Bend over LCD pin 10, or use LCD's cable set per wiki instructions	

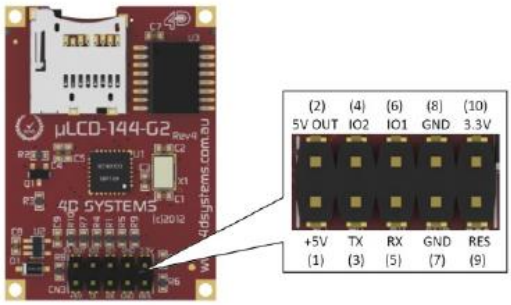


Figure 1: These are the LCD pin connections needed for this assignment. Please note that there are some pins moved between this figure and the *mbed* cookbook wiki demo code. Note that if you choose to use the cable, the RX and TX pins are swapped on the cable connector label (per wiki).

6. Create a new program in your compiler environment and type in the following code. You will need to copy the uLCD_4DGL folder from the skeleton code project in third step into this new project. Cut and paste can be used in the compiler's left column that shows project source files. You can also use the import library link found on the LCD wiki page.

```
#include "mbed.h"
#include "uLCD_4DGL.h"

uLCD_4DGL uLCD(p28, p27, p29); // create a global lcd object

int main() {
        uLCD.printf("\nWhat's up mbed!\n");
}
```

7. Compile this program. On some machines, this will automatically download into your *mbed* if it is connected to your computer; however, on other machines the file might be in your download folder. In this case, all you need to do is drag it to the *mbed* drive, and it will be automatically downloaded to the board.

8. Pressing the reset button on the chip will automatically run the program with the most recent timestamp that is downloaded to the board. Do this and you should see your first message on the LCD display!

Instructions for Part 2 (10%): "Bouncing Ball"

Please input the following program and run this on your *mbed* system with the LCD display. You might play around with changing the ball size and colors. There is a lot of documentation on drawing different objects on the *mbed* website.

```
// uLCD-144-G2 demo program for uLCD-4GL LCD driver library
#include "mbed.h"
#include "uLCD_4DGL.h"

uLCD_4DGL uLCD(p28, p27, p29); // serial tx, serial rx, reset pin;

int main()
{
        uLCD.display_control(PORTRAIT);
        uLCD.cls();
        uLCD.printf("Bouncing Ball");
        uLCD.baudrate(BAUD_3000000); //jack up baud rate to max for fast display
        wait(1.0);

        //Set up initial conditions
        float fx=50.0,fy=21.0,vx=0.4,vy=0.3;
        int x=50,y=21,radius=4;
        uLCD.background_color(BLACK);
        uLCD.cls();

        //draw borders
        uLCD.line(0, 0, 127, 0, WHITE);
        uLCD.line(127, 0, 127, 127, WHITE);
        uLCD.line(127, 127, 0, 127, WHITE);
        uLCD.line(0, 127, 0, 0, WHITE);

        for (int i=0; i<1500; i++)
        {
                //draw ball
                uLCD.circle(x, y, radius, RED);
                //bounce off edge walls and slow down a bit
                if ((x<=radius+1) || (x>=126-radius)) vx = -.95*vx;
                if ((y<=radius+1) || (y>=126-radius)) vy = -.95*vy;
                //erase old ball location
                uLCD.filled_circle(x, y, radius, BLACK);
                //calculate new ball position
                fx=fx+vx;
                fy=fy+vy;
                x=(int)fx;
                y=(int)fy;
        } //end for loop
}//end main
```

Instructions for Part 3 (10%): "Digital Thermometer"

The TMP36 wiki page link is provided below that shows how to connect the TMP36 sensor and read the temperature using C/C++ on the mbed module. BE CAREFUL because it looks just like the 2N3904 transistor in the kit, so check the tiny marks on the case's flat area for the part number. This assignment will verify your hardware connections and sensor operation before trying your next mbed lab.

1. In previous sections you got your LCD working, please leave this hooked up!
Now hook up the temperature sensor using the information on the wiki page at:
http://mbed.org/users/4180_1/notebook/lm61-analog-temperature-sensor/

2. Create a new program and type in the following code.

```
#include "mbed.h"
#include "uLCD_4DGL.h"

uLCD_4DGL uLCD(p28, p27, p29); // serial tx, serial rx, reset pin;

//Setup a new class for TMP36 sensor
class TMP36
{
public:
        TMP36(PinName pin);
        TMP36();
        float read();
private:
        //class sets up the AnalogIn pin
        AnalogIn _pin;
};

TMP36::TMP36(PinName pin) : _pin(pin) {} //This is an initializer list … more to come in class
// _pin(pin) means pass pin to the AnalogIn constructor

float TMP36::read()
{
        //convert sensor reading to temperature in degrees C
        return ((_pin.read()*3.3)-0.500)*100.0;
        //read() function returns a normalized value of the voltage from 0 to 1 as a float
}

//instantiate new class to set p15 to analog input
//to read and convert TMP36 sensor's voltage output

TMP36 myTMP36(p15);
```

```
int main()
{
	float tempC, tempF;
	while(1) {
		tempC = myTMP36.read();
		//convert to degrees F
		tempF = (9.0*tempC)/5.0 + 32.0;
		//print current temp
		uLCD.cls();
		uLCD.printf("%5.2f C %5.2f F \n\r", tempC, tempF);
		wait(.5);
	}
}
```

Instructions for Part 4 (20%): "Jazzy Tunes"

In the fourth part of this lab you will build a basic 3-key musical keyboard. Do not take off the other components! The point of this exercise is to build up and verify your components to help with your future *mbed* lab. I would advise that you try to get the keys working first with some of the LEDs and then add functionality with the speaker. I will not spell out everything explicitly here, but instead I will require you to get information from the *mbed* website.

Pushbuttons

Read the pushbutton wiki page at http://mbed.org/users/4180_1/notebook/pushbuttons/ and watch the videos for additional help using pushbuttons with *mbed*. Small pushbuttons are available for use on your breadboard. I would like for you to look at the constructor calls in the sample code to determine the pin connections to the mbed board.

Speaker

WARNING! WARNING! WARNING WILL ROBINSON!!!

THE SPEAKERS ARE VERY DELICATE. DO NOT PULL UP ONCE IT IS ON THE BREADBOARD. YOU MUST EASE IT OUT BY PRYING IT OFF FROM UNDERNEATH THE SPEAKER. IT IS TRICKY. BE CAREFUL.

Use the driver transistor and speaker to make tones when each button is pressed. See the speaker wiki page for additional hardware and software help using speakers with mbed which is found at https://mbed.org/users/4180_1/notebook/using-a-speaker-for-audio-output/

I would like for you to look at the constructors in the sample code to determine the pin connections to the *mbed* board.
1. Please leave all previous components hooked up on your board (i.e. LCD and temperature sensor)
2. Now hook up three push buttons using the information on the wiki page at http://mbed.org/users/4180_1/notebook/pushbuttons/

I would suggest testing the code on the following page without the PlayNote functions to see if the push buttons work with the LEDs built into the mbed board.

3. Now hook up the speaker and transistor driver using the information on the wiki page found at https://mbed.org/users/4180_1/notebook/using-a-speaker-for-audio-output/

4. Create a new program and type in the following code. You will need to copy the Speaker.h file over from the skeleton code that you downloaded to your mbed account in part 1 of this lab.

```
#include "mbed.h"
#include "Speaker.h"
#include "PinDetect.h"

DigitalOut myled1(LED1);
DigitalOut myled2(LED2);
DigitalOut myled3(LED3);
DigitalOut myled4(LED4);

PinDetect pb1(p16); //you can use different pins
PinDetect pb2(p17); //you can use different pins
PinDetect pb3(p18); // you can use different pins

// setup instance of new Speaker class, mySpeaker using pin 21
// the pin must be a PWM output pin
Speaker mySpeaker(p21);

//-------------------------------------------------------------------------------------------------
// Callback routine is interrupt activated by a debounced pb1 hit
// That is … this code runs with interrupt is generated by first button press

void pb1_hit_callback (void)
{
        myled1 = !myled1;
        mySpeaker.PlayNote(200.0,0.25,0.1);
}

//-------------------------------------------------------------------------------------------------
// Callback routine is interrupt activated by a debounced pb2 hit
// That is … this code runs with interrupt is generated by second button press
void pb2_hit_callback (void)
{
        myled2 = !myled2;
        mySpeaker.PlayNote(400.0,0.25,0.1);
}

//-------------------------------------------------------------------------------------------------
// Callback routine is interrupt activated by a debounced pb3 hit
// That is … this code runs with interrupt is generated by third button press

void pb3_hit_callback (void)
{
        myled3 = !myled3;
        mySpeaker.PlayNote(800.0,0.25,0.1);
}

//-------------------------------------------------------------------------------------------------
int main()
{
        //setup push buttons
        pb1.mode(PullUp);
        pb2.mode(PullUp);
        pb3.mode(PullUp);
```

```
        // Delay for initial pullup to take effect
        wait(.01);
        // Setup Interrupt callback functions for a pb hit
        pb1.attach_deasserted(&pb1_hit_callback);
        pb2.attach_deasserted(&pb2_hit_callback);
        pb3.attach_deasserted(&pb3_hit_callback);
        // Start sampling pb inputs using interrupts
        pb1.setSampleFrequency();
        pb2.setSampleFrequency();
        pb3.setSampleFrequency();
        // pushbuttons now setup and running

        while(1)
        {
                myled4 = !myled4;
                wait(0.5);
        }
} //end main
```

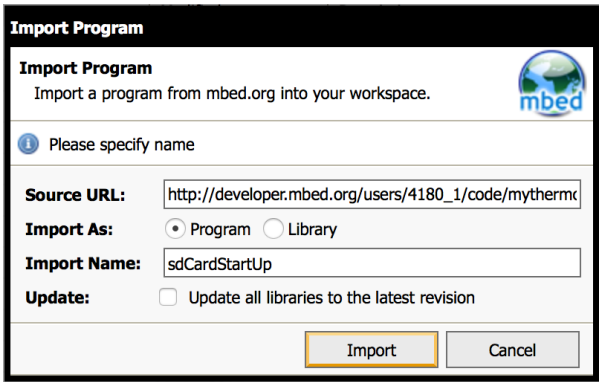
Instructions for Part 5 (20%): “Hello Micro SD Card”

Part A: Writing Text ToYour SD Card

In this part, you will need to add the micro SD card to your setup.

To make sure that you have the correct libraries to get your SD card working, I would suggest that you start a new program by downloading this skeleton code from the following link:
http://mbed.org/users/4180_1/code/mythermostat/

Click on the orange "Import into Compiler" on the right-hand side of the window. The following pop-up window will appear in your browser. I suggest populating the pop-up window in the following way.



Now you can simply edit the main.cpp that is in this directory to include the code listed below.

The micro SD card (http://mbed.org/cookbook/SD-Card-File-System) or a USB flash drive (<2G FAT 16) can be used for storage. For this lab you will use the SD card, and the pin connections to your mBED are illustrated in the previous link. Please use the constructor call in the below sample code that WRITES a simple message to file on your SD card to determine the pin connections that you need. The microSD card can be put into the larger SD card adapter to read the data files on a PC if you would like. An example file that you might use to verify that your SD card is working is given below.

(code on next page)

```
#include "mbed.h"
#include "SDFileSystem.h"
#include "uLCD_4DGL.h"

SDFileSystem sd(p5, p6, p7, p8, "sd");

uLCD_4DGL uLCD(p28, p27, p29); // serial tx, serial rx, reset pin;

int main() {
        uLCD.printf("Hello Micro SD Card!\n");
        mkdir("/sd/mydir", 0777);
        FILE *fp = fopen("/sd/mydir/sdtest.txt", "w");
        if(fp == NULL) {
                uLCD.printf("Error Open \n");
        }
        fprintf(fp, "I love ECE2036");
        fclose(fp);
}
```

PLEASE NOTE: Writing files without an SD card plugged in can result in the mBED "blue LEDs of death".

Part B: Reading the text file off your SD Card

If you have an SD port on your laptop, you should be able to read the file that you just created on your SD card. You can do this to demo to the TAs that you successfully wrote to the SD card. However, if you do not, then you can use the following code to have your mBED system read the information off the card so that you can demo to the TA.

```
#include "mbed.h"
#include "SDFileSystem.h"
#include "uLCD_4DGL.h"
#include <string>

// use class to setup microSD card filesystem
SDFileSystem sd(p5, p6, p7, p8, "sd");

// use class to setup the Color LCD
uLCD_4DGL uLCD(p28, p27, p29); // create a global uLCD object

int main()
{
        string inputString;
        uLCD.printf("Content of sdtest.txt is: \n");
        FILE *fp = fopen("/sd/mydir/sdtest.txt", "r");
        if(fp == NULL) {
                uLCD.printf("Error Open \n");
        }
        else
        {
                while (fscanf(fp,"%s", inputString)!= EOF) //reads in a string delineated by white space
                {
                        uLCD.printf("%s ", inputString.c_str());
                }
        }
        fclose(fp);
}
```

Instructions For Part 6(30%): Triple Axis Accelerometer

1. Please leave all previous components hooked up on your board (i.e. LCD and temperature sensor)
2. Please connect the pins on mBED to the pins on your MMA8452Q according to the following table. You can also find more information about this accelerometer at the following URL: https://os.mbed.com/components/MMA8452Q-Triple-Axis-Accelerometer/

MMA8452Q Pins	mBED Pins
Vcc	Vout (3.3V)
SDA	P9
SCL	P10
I1	P26
I2	P25
GRD	GRD

3. Download and run the following program at the following URL. This will include the libraries that you need to get the accelerometer working.
https://os.mbed.com/users/jd7142128/code/AccStarter/

4. Please demo the operation of this code to the TAs. There is a way to activate a musical component to this lab. Can you figure this out from the code? Wow the TAs and they might give a few extra points!