ECE 4180 Lab 2 – Everything I/O with mbed - part 2

Due Date: Even: Wednesday, September 24
Odd: Thursday, September 25

Names:_____ _____

Item	Lab Demo	Extra Credit	Extra Credit
4180 LAB 2	n/a	n/a	n/a
Analog Input		n/a	n/a
Analog Output		n/a	n/a
I^2C		n/a	n/a
RS232 Serial		n/a	n/a
Ethernet Networking			**3%**
PWM Servo or DC motor			**2%**
uLCD Graphics Display			**1%**
MicroSD Card File I/O		n/a	n/a
Speaker Audio Output		n/a	n/a
Hardware Breakpoint Demo		n/a	n/a
Robot Dance	n/a		**2%**
USB	n/a		**3%**
IR and RF data transmission	n/a		**2%**
Relay or Power Switch Tail	n/a		**1%**
Internet of Things LCD gadget	n/a		**2%**

TA Signoff: _____

Note: Try to keep the stations organized. Once you are done with a part return it to the TA or leave it at the station. If the cabinets are open, return them to the correct yellow labeled bin.

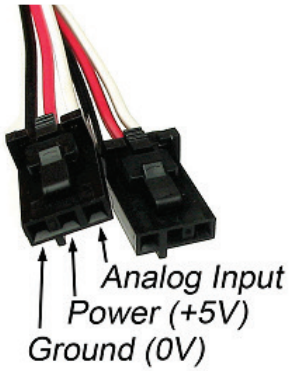
Please Note: The large white breadboards now have 5V 4A switching power supplies connected via the DB-9 connector on the back as seen below. The toggle switch on the top left can turn the connection to the power supply on/off along with a status LED. The power supply is connected to the binding posts on the bottom right of the board. This power supply can easily power the MBED with all of the components in this second lab**, except for the large DC motor which might require more than 4A if it is stalled**. If you use both USB 5V for mbed and the second power supply for the motor, be sure to connect ground wires only between the two 5V supplies (and not the +5V).

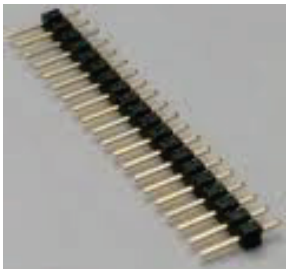
ECE 4180 Laboratory Assignment 2

For lab 2, we will be using several modules with surface mount ICs mounted on small breakout boards that will allow them to plug into a protoboard. These boards are stored in yellow bins in the lab cabinets, if they are not out. You can check off parts one at a time or demo the entire lab using the big everything loop at the end assuming you still have everything hooked up.

Part 1 Analog Input and Output (15%)

Send out a sine wave on the analog output pin. Use 20+ samples per cycle and send out two complete cycles. Bias and scale the analog output so that it uses the full analog output range. It should output two complete cycles each time through the Part 1loop. Check the output with a scope and note the frequency. Capture the scope image and hand it in at checkout.



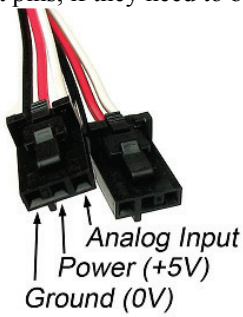


Read in analog data from a Sharp IR distance sensor. These have a strange three pin cable (JST-metric) and you can plug jumper wires into the end of the connector to connect to the protoboard. It is probably easier to use it with the Phidgets PCB IR modules in the lab with the cable shown above since this cable has standard .1 spacing and the PCB has a filter. Tie it to 5V, GND and an analog in pin as seen above . We have some of the long SIP header pin strips that can be used to plug these three wire cables directly into the protoboard as seen in the right image.

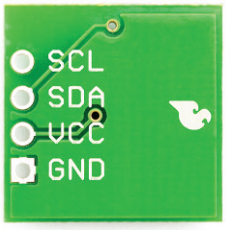
For info on the sensor pin out and the distance and voltage relationship, see the Sharp IR sensor's data sheet at http://sharp-world.com/products/device/lineup/data/pdf/datasheet/gp2y0a21yk_e.pdf. Output the range data from the Sharp IR sensor using a bar graph type display on the mbed's four built-in LEDs (LED1-4, See *DigitalOut*). Write the C code and add it to the program loop. Check out the *AnalogOut* and *AnalogIn* APIs in the handbook. You should be able to move your hand back and forth above the IR sensor and see the LED bar graph display change.

Check-off: Show the TA the IR sensor bar graph that uses the built-in LEDs.

As an alternative to the IR sensor, you can also use one of the small Maxbotix EZ analog Sonar modules, but it would tie into 3.3V for power (5V would saturate an mbed analog in pin that is looking for 3.3V max!). http://www.maxbotix.com/uploads/LV-MaxSonar-EZ1-Datasheet.pdf has the pin out info. The 3-wire small cable for it needs to be soldered to the PCB, if one is not already attached. With harsh use in the lab, the wires always seem to break off. Make sure wires are connected to the correct pins, if they need to be resoldered.



Part 2 I²C bus and USB virtual com port (10%)



The compass module in the lab is the HMC6352 I²C digital compass module made by Honeywell (http://www.sparkfun.com/products/7915) for which the example code and library can be found here: http://mbed.org/cookbook/HMC6352-Digital-Compass.

Read carefully, and **note that two 2.2-47K pull up resistors are required for the I²C bus since it uses the older open collector style outputs and not tri-state for the bus**. Add the C code for this sensor to the program loop. Open up the compass library code and note the *I2C* calls in *HMC6352.cpp*, used to control the module and transfer the data. Some I2C breakouts already have the pullups and some do not, the schematic must be checked. The I2C code hangs if there are no pullups. The compass has the pullups on the board.

The output from *pc.printf* is sent using a USB virtual com port to the PC. The virtual com port uses the same USB cable from the mbed module, so no additional cables are needed. Use a terminal application program such as Hyperterm or Realterm on the PC to display the data. More details are available at http://mbed.org/handbook/SerialPC. The mbed serial driver locks into the module's unique serial number so you have to select the right one to connect to from the list of serial ports. If you noted the serial number earlier when setting up your account, find the entry for it. If you are working at home or on your laptop, a special virtual com port driver is required and it locks to each board's serial number. Each mbed board needs it's own install.

Check-off: Show the TA the compass correctly displaying bearings through the I²C interface the USB virtual com port.

New Optional Parts kit:

The compass module was a bit pricey ($35) and not currently available, so use your I2C temperature and humidity or the accelerometer sensor from the kit instead to print the temperature and humidity data, or the G forces on each axis to the serial port. They do have a new more accurate compass for $149 in stock, if you really want your own electronic compass. Mbed cookbook code for the temperature and humidity sensor in the kit is available at https://mbed.org/users/hwing91/notebook/htu21d---temperature-and-humidity-sensor/. This code outputs to the color LCD, so modify it to print to the PC using the USB virtual com port as in the compass instructions above. This helps in the next part of the lab which will use a different serial output to the PC. For the accelerometer option, a code library is available at http://mbed.org/users/dagronlund/code/AccelSensor/file/7dd118f48b1b/AccelSensor.cpp (the hello world code example and a nice cookbook page for this one is not around yet – this might make a nice lab 3 project for someone). A new alternative accelerometer library someone just found might be easier to use at http://mbed.org/users/ashleymills/code/MMA8452/file/dfb0f6a7a455/MMA8452.cpp. with example code here: http://mbed.org/users/ashleymills/code/MMA8452_Test/ that shows how to use it.

Part 3 Real RS232 Serial (10%)

Mbed	adapter
3.3V	VCC
GND	GND
RX	TX
TX	RX

Many I/O devices still require the older RS232 standard serial interface. In addition to the virtual serial com port, output the compass heading info from a "real" RS232 serial port to the PC. Attach a serial cable to the PC and display it using a terminal application program running on a real COM port (i.e., not the USB virtual com port). RS232 voltage level conversion (i.e., +3 and –3 voltage) is required and some very handy small RS232 SMD adapters from Sparkfun http://www.sparkfun.com/commerce/product_info.php?products_id=449 are available in the lab. If these were not available, you would need to use an IC like a MAX232 and some caps on the protoboard to adjust the voltage levels.

Be aware of possible null modem serial cable issues. There are some small null modem adapters and gender changers in the lab as seen in the image above on the right, if you need one. Add this output code to your C code to the program loop. There are "real" serial port code examples in the handbook and cookbook and check out the *Serial* API. You cannot just switch the wires on the mbed's *TX* and *RX* pins to make a null modem connection (i.e., a wire is bidirectional, but the voltage conversion circuit is not). **If you are outputting from the MBED the red RX light should blink, while if you are reading the green TX light should blink on the Sparkfun adapter**.

Check-off: Show the same functionality in Part 2, but use the RS232 serial port instead of the virtual serial com port.

New Optional Parts kit:

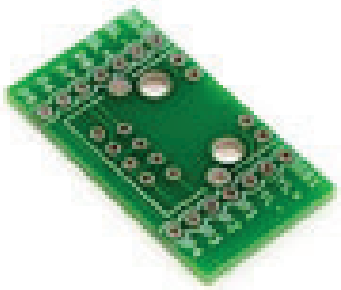
The PCs in the labs have a com port and serial cables are available in the lab.

If you want to work at home and do not have a PC serial cable the TA can let you borrow one from the lab. Gender changers are also available in the lab if needed. A few PC cables already swap TX and RX (i.e., null modem cable) and do not need a null modem adapter. If would be too easy if they just marked which type of cable it is somehow! There is a grey tub of these PC serial cables in the back left corner of the lab.

If your laptop or PC at home does not have a serial port, we have some USB to serial converters you can borrow in the lab. These need a device driver installed and it probably will not automatically do it. Windows 8 does not like the low-cost USB to serial converters in the lab – according to the web blogs they have a counterfeit USB chip from China and the new Windows 8 drivers check for it and stop working. If you have Windows 8, and still want to work at home, real USB to serial cables that work with Windows 8 are available for around $15. Make sure it says that it "works with Windows 8," on the box. Be careful, the web blogs make it sound like there are more counterfeit than real ones out there! The two companies (Prolific and FTDI) that make the real USB to Serial chips have web sites with links to drivers and some places to buy a "real" cable.

Part 4 Networking: Ethernet (15%)

Mbed	adapter
TD+	P1
TD-	P2
RD+	P7
RD-	P8



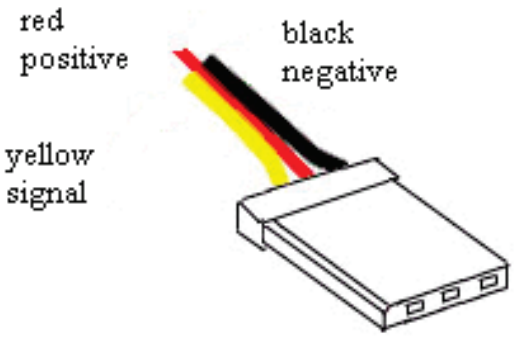
Demonstrate one of the mbed's cookbook Ethernet projects. A small Ethernet breakout module is available in the lab and the mbed web site has some networking demo code. For debugging purposes you can detect packets with Wireshark or Ethereal. Magjack connectors from different companies can have different pinouts. The ones we have are shown in the table. http://mbed.org/handbook/Ethernet is probably a good test project to try first to see if your connection to the network jack is correct. To learn more about networking on the mbed, http://mbed.org/handbook/Networking has a lot of useful information and includes the most recent officially supported libraries. Mbed recently deprecated many of their libraries, so if you're running into a lot of errors, this might be your problem.

Check-off: Demonstrate an HTTP client executing a GET request. You can use the USB virtual com port to display useful connection information.

Note: For DCHP to work and get an IP address for mbed, the mbed module's NIC address must be registered with ECE. Most have been registered, but if you grab a new one out of a box or bring your own from the parts kit it might get a timeout error during DHCP. If you have a laptop with WiFi and a network jack you could setup a network bridge connection for the mbed module to get around this. It takes a couple of days with email delays to computer support people to add a missing NIC address to the DHCP servers DHCP enable tables and it is just not a good solution for all of the new student owned mbeds. They are also running out of IP addresses in Van Leer and need new equipment to be able to fix it. But the new equipment would require dealing with the asbestos in the ceilings!

For people with the new mbed inventor's kit, in newer buildings like Klaus that come up with a GTLawn page for unknown MACs, it might be possible to use the MAC address to enable a temporary guest account or if your laptop connects with Wi-Fi, you might be able to bridge to network jack for internet access. Using GTother it is possible to setup a guest account for another device such as mbed. Also it is possible to setup a network bridge on a laptop with both WiFi and an Ethernet jack. A network bridge is the recommended method for mbeds without registered MAC addresses. There seem to be some strange DHCP bridge issues with Windows 8.1 and extra credit is available for anyone that can figure it out. You may need to use a static IP on mbed instead of DHCP in Windows 8.1 as suggested in the cookbook page.

Part 5 PWM: Servo position control or DC motor speed control (10%)



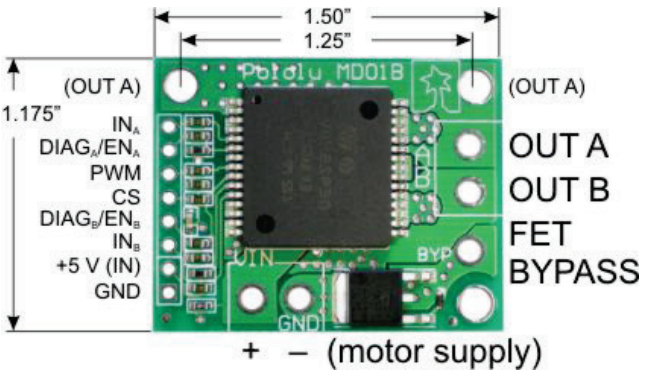
Check-off: Move an RC servo using PWM on the mbed or do a speed and direction control demo on a DC motor using PWM. A new servo wiki page is available in the cookbook.

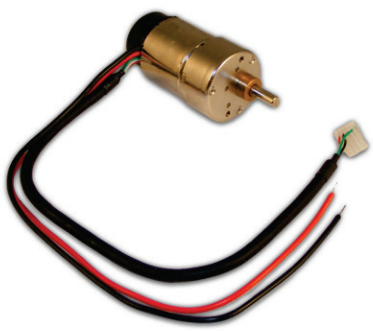
Servos have a three wire connector with 5-6V DC, GND and the PWM control signal. All of the RC servos have the three wire connectors and 4.5-6V DC is typically the middle pin, but the different brands move the three pins around and use different wire color codes. Each company does this to try to lock users into their RC equipment. The info on which pin is which can be hard to find. There is even a company that makes conversion cables! Here is a handy pin out table with some cable pictures like the one above:

http://www.horrorseek.com/home/halloween/wolfstone/Motors/svoint_RCServos.html#ServoConnectors

The mbed module has built-in power protection that trips at 460MA, so an **external power supply for servos or a DC motor must be used** since a servo's internal DC motor or a regular DC motor has a large in rush current at startup (or if it stalls). Don't forget to tie the two power supply grounds together at one point (but not 5V!).

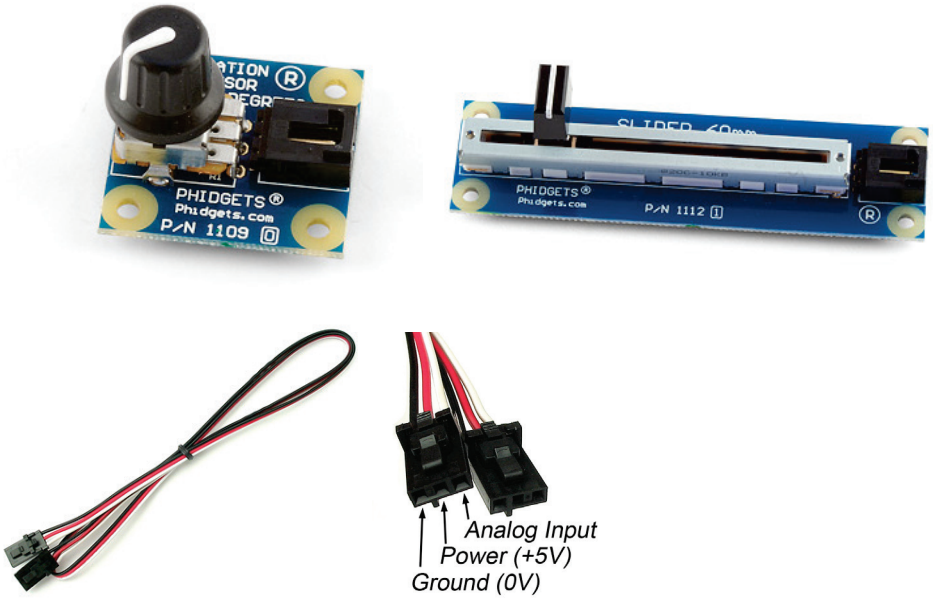
If you use a DC motor to demo speed control with PWM you will need to also add and H-bridge driver circuit. There are some H-bridge driver modules in the lab. There are servo and DC motor examples in the cookbook motor section. In the handbook, check out the *PWMOut* API. The H-bridge module below is from Polou and is used in automobile car seats. The geared DC motor with a built in encoder is from RoboticsConnection. It can handle up to 7.2V DC. For pin functions, see the datasheet on page 7 at http://www.pololu.com/file/0J51/vnh3sp30.pdf.





Mbed	H-bridge	Power supply	Motor
Vu	+5V (IN)		
GND	GND		
rev	IN_B		
fwd	IN_A		
Pull high	EN_A		
Pull high	EN_B		
pwm	PWM		
	+ motor supply	+ power supply	
GND	- motor supply	Ground of power supply	
	OUT A		Motor wire 1
	OUT B		Motor wire 2

The position of the servo, or the DC motor's speed and direction must be controlled using a potentiometer setup as a voltage divider circuit connected to an mbed analog in pin. We have these Phidgets modules mounted on small PCB with a three wire cable available in the lab. You can decide to go rotary or slider. Use 3.3V for power on these modules instead of 5V as shown in the image. The motor driver module shown above supports the brake feature. Use the DIP switch input or a Phidgets analog touch switch to turn on the motor brake feature. **Please make sure you hook up the polarity for the H-Bridge correctly (motor supply in the image above), the polarity is noted on the bottom of the PCB. If the polarity is reversed the H-Bridge will be permanently damaged.**



New Optional Parts kit:

Use one of the small DC motors in the robot kit and the dual H-bridge driver breakout in your kit. A cookbook page for the robot is available at: https://mbed.org/users/4180_1/notebook/sparkfun-magician-robot-base-kit/ . It is not required to hookup both motors or assemble the full robot kit unless you want to. If you want to work at home and do not have a potentiometer, two pushbuttons can be used to increase and decrease motor speed. With a bit more work, the touch keypad could also be used. A code example for the touch keypad can be found at http://mbed.org/users/4180_1/notebook/mpr121-i2c-capacitive-touch-sensor/. The power supply is a bit limited over the USB cable and you might need a bit more current. If you notice the power LED blink when the motor turns on this is likely the problem. If you have an old AC wall wart around that outputs 4-6VDC, it could be used to power the DC motor only. Don't forget to hook up the external power supply to mbed's ground (but not the two 5V pins). Another solution would be to use the 4AA battery pack in the robot kit to power your motor.

Part 6 TTL Level RS232 Color Graphics LCD display (10%)

Color Graphics LCD Module

CAUTION

Power on pin 1 (+5V) on the Color LCD in your parts kit **uses only the 5V mbed pin VU** (i.e., **not 3.3V – mbed pin Vout**) and pin 7 (GND) is connected to the mbed GND pin. Always double check power pin connections before turning on power for the first time – if you get them wrong it might burn out the device! See the Color LCD wiki page for additional help using the Color LCD with mbed. If you want to plug the LCD directly in a breadboard, it is necessary to carefully bend over one pin as shown on the wiki page. The LCD requires several wires and if it is placed near the mbed pins used, the breadboard setup will be easier. Note that the mbed pins used (27, 28, 29) are on the right side of the mbed module and these pins are a bit different than the wiki page example (LCD code can use any of mbed's three Serial outputs just by changing constructor pin number arguments), but use them to be compatible with the embedded systems you will build in labs that follow to have space for all of the parts added later. It turns out that it is more common to have more parts and the left side of the mbed module and this setup will leave space for them.

On your protoboard, please put the LCD on the side of the MBED chip where you have the pins p27-29. See Tablein the cookbook page for the pin connections needed.

Create a new program in your compiler environment and type in the following code. Remember … you will need to copy uLCD_4DGL.h into this new project. Use the import library link found on the Color LCD wiki page referenced above.

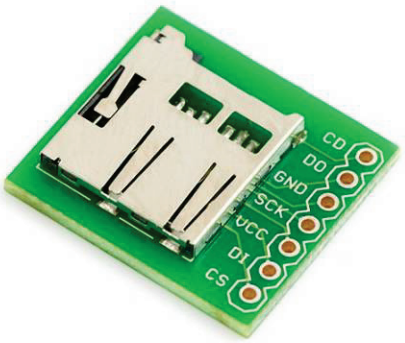
```
// Hello World! for the TextLCD

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

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

int main() {
   uLCD.printf("\nHello World!\n");
}
```

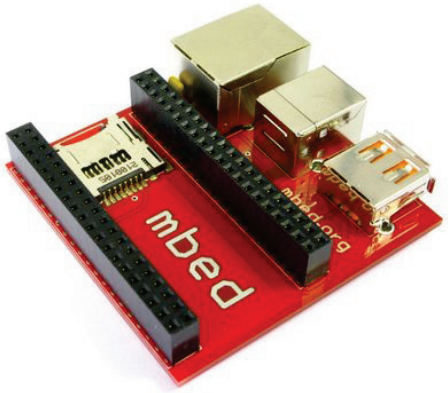
Check-off: Show the TA the working demo code.

Part 7 SPI bus: microSD card file system (10%)

Add a microSD or SD file system. SD cards are interfaced using an SPI bus. MicroSD breakout boards are available from Sparkfun and a larger mbed breakout setup from the UK. The cookbook has code examples. Note that you will probably need to use a microSD card brand and size that has been noted as working OK in the mbed cookbook and forums (2G to 32G?). Write a program to create and write a file that contains "Hello SD file world" closes it, and then opens it and reads the file back on the PC using the USB virtual com port. (Hint: the local file system only supports 8.3 filenames -- http://en.wikipedia.org/wiki/8.3_filename). Leave the SD card setup for possible use in the next part. Many laptops have a built-in SD card reader that will work with the SD to uSD adapter.

Check-off: Show the TA the working file I/O program.





Part 8: Add the speaker (10%)

Check-off: Use a speaker and playnote() to play the first 17 notes of the Rambling Wreck Song, or play an SD wave file of SiFi theme music or a well known SiFi sound effect. See the speaker wiki page.

Part 9: Use Breakpoints in the offline compiler (10%)

For software development, the easy-to-use mbed cloud compiler is typically used. Projects from the cloud compiler can also be exported to the offline compiler. Breakpoints will work in the offline compiler with a firmware update for the mbed module. The offline compiler takes a bit more time and effort, but it supports breakpoints for debugging complex programs. For the offline compiler, download the free demo version (<32K code size) of the ARM/Keil MDK compiler. To enable the offline compiler's full features, you must start the compiler and then connect over the network using VPN with your GT password to Georgia Tech's FlexLM license server. VPN should not be needed on and ECE lab PC. Use **File->License Mangement ->FlexLM**. Then set the FLEX server to **8224@ECEWINSRV1.ECE.GATECH.EDU** and the full version will be enabled.

Check-off: Demo breakpoints using this Cylon LED project code which can be imported to the cloud compiler.

Extra Credit Options for Lab 2 (some may require additional parts from lab TA)

(1%) Display the current time from the mbed's real-time clock on the LCD used in Part 5 using larger red characters in the middle of the LCD. See the *time* link in the handbook. Update the time every 200 mS.

(2%) Demo the other device (servo or DC motor) not used in your earlier Part 5 demo.

(2%) If you have the optional parts kit**,** build the robot kit, hookup both DC motors to the H-bridge and make a demo that makes the robot move around in a pattern such as forward a foot, backwards a foot, a full rotation CW and then CCW. For those without the optional kit, some robot kits are also available in the lab.

(3%) Demo the HTTP Web Server using the mbed's flash file system for web page storage.

(3%) Demo one of the mbed cookbook's two USB projects (to demo reading from and writing to a USB device) or the one that can be found at: http://mbed.org/users/chris/programs/MSCUsbHost_FULL/5ymp5/docs/files.html. (Hint: the local file system only supports 8.3 char filenames -- http://en.wikipedia.org/wiki/8.3_filename)

(2%) Demo IR and RF data transmission using the setup found at: http://mbed.org/users/4180_1/notebook/ir-and-rf-remote-controls/

(1%) Demo one of the relay modules or a Power Switch Tail -- http://mbed.org/users/4180_1/notebook/relays1/

(2%) Demo one of the Internet of Things LCD gadgets similar to these at Nokia Color Graphics LCD panel -- http://mbed.org/cookbook/Internet-of-Things---LCD-Gadgets, but using the newer Color Graphics uLCD used earlier in the lab. The Google weather API not free anymore, so that one would require finding another weather web API such as Yahoo and rewriting the code. There is a student design project posted in the cookbook project pages that already has this code for the clock.

(**2%**) Extra credit for the first two teams to complete the lab using the optional parts kit – since the instructions are new for this option.