MHIT 603 Assignment 3: Arduino and Processing

Motivation

In class you've been learning about Arduino and Electronics. In this assignment, you will be working with the Arduino, using it to communicate with and control a Processing application and vice versa.

Assignment Description

There are three parts to the assignment, one focused on Arduino to Processing communication, one focused on Processing to Arduino communication, and one focused on communication in both directions. The first two parts make up the bulk of the grade (80%), while the last part is more of an "extra for experts" task. You will receive credit for the parts you are able to complete.

A zip file is provided with example code for communication from Arduino to Processing, and Processing to Arduino. You may choose to use build off these files, use other files provided in class, or write your own from scratch.

You should submit a separate zip file of the Arduino and Processing code for each part of the assignment, and include a photograph and/or instructions on how to set up the electronic components with your Arduino.

Marks will be given for completing the task, as well as the quality of the code you write. You should include comments which explain your code, and try and make your code easy to understand. Some of the marks awarded will be given for doing more than the minimum requirement, so if you have time you should try to add some extra flair to your designs.

The assignment is due 11:55pm on Sunday 11th of October.

This in an individual assignment, your work must be your own, although you are allowed to discuss high level topics with others (i.e. what functions to use to achieve a certain task, etc).

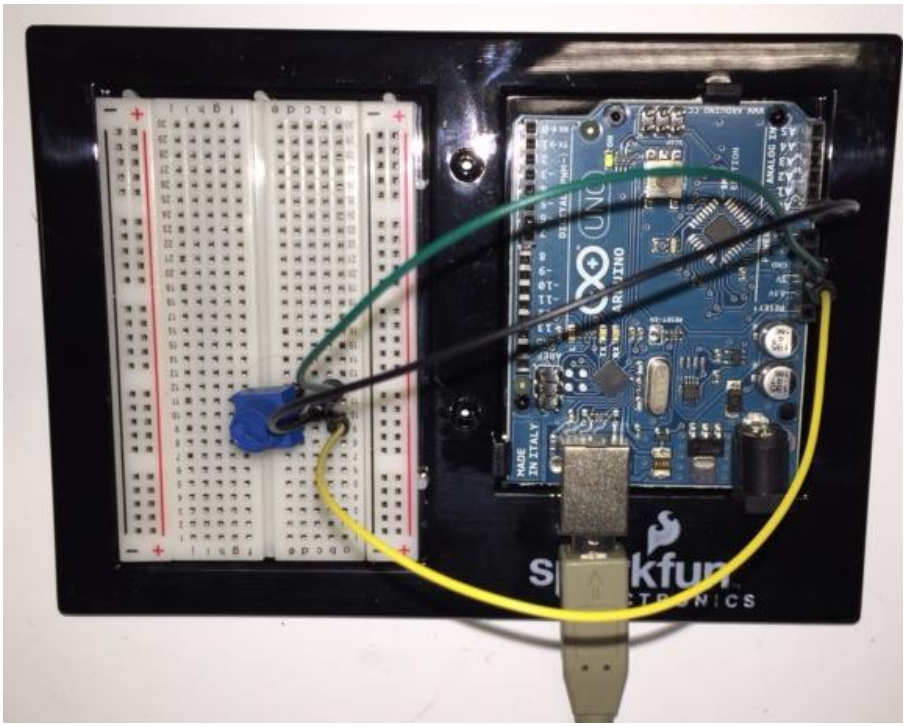
The parts are as follows:

Part 1: Arduino to Processing (40%)

In this part, you will be using your Arduino to control a processing application. In the sample source code, you can find the Arduino and Processing source code in the "Arduino to Processing" folder.

The example source code assumes you have the middle pin of a potentiometer (variable resistor) connected to A0 on the Arduino, with the other two pins connected to GND and 5V respectively, as shown in the image below. When you upload the Arduino code to the board and run the processing example, you will see it prints out the current value of the potentiometer in the processing console area. The data is sent in packets with the format `<AnalogValue>`, and all the delimiting and buffering is

handled by the processing function getValidPacketsFromSerialPort(), which returns a string array of every valid packet received since the last call.



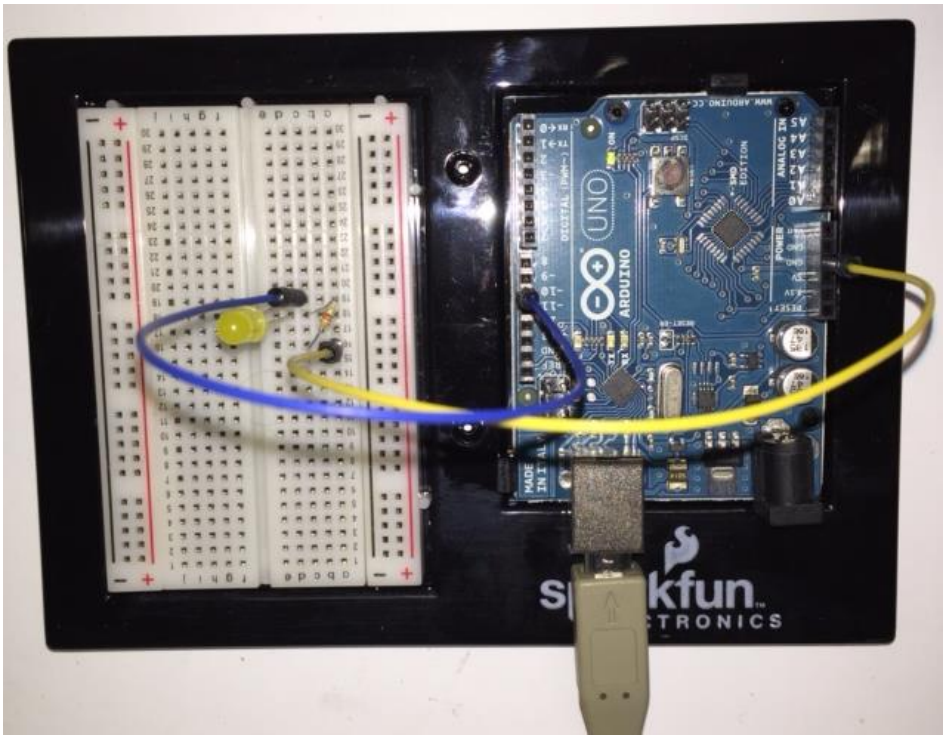
Your task in this part of the project is to create a custom controller for a 3D model viewer in processing. Your controller should have at least 1 potentiometer which will be used to rotate the model, and 1 button which is used to toggle between two models. You should look at the Arduino example “Basics->Digital Read Serial” to see how to read a push button input, and you could use something like the “Topics->Textures->TextureCube” and “Topics->Textures->TextureSphere” as the example code for the processing graphics. You should think about how you want to structure your communication protocol, perhaps something like `<AnalogValue,DigitalValue>`, which you will then need to parse.

Part 2: Processing to Arduino (40%)

In this part, you will be controlling your Arduino with a processing application. In the sample source code, you can find the Arduino and Processing source code in the “Processing to Arduino” folder.

The example source code assumes you have the positive leg of an LED (the leg on the round side of the LED) connected to D10 on the Arduino, with the negative end of the LED (flat side of the LED) connected to a 330 ohm resistor and then to GND, as shown in the image below. When you upload the Arduino code to the board and run the processing example, you will see the brightness of the LED can be changed by the X position of the mouse in the processing window. The data is sent in packets with the format `<MOUSEXPOSITION>`, and all the delimiting and buffering is handled by the Arduino function

getMostRecentValidSerialPacket (), which returns a string with the most recent packet which has been received by the Arduino.



Your task in this part of the project is to create a decimal to binary converter for the numbers 0-7. You should have 3 LEDS on your Arduino which light up to show binary values, i.e. 0 dec = 000 binary, 7 dec = 111 binary. You should detect when someone presses one of these numbers in your processing application, and then light up the corresponding LEDs on the Arduino to show what the binary value of that number is. It is up to you how to encode the packets.

Part 3: Multi-directional Communication (20%)

In this part of the project, you should combine the Arduino->Processing and Processing->Arduino code, to create a system where the Arduino can control the Processing application, and the Processing application will show feedback on the Arduino. You are free to do whatever application you want for this. 15% of the marks will be given for getting multi-directional communication working, with the remaining 5% for how interesting your application is.

Marking Schedule

This assignment is worth 10% of your grade for MHIT 603. Marks will be determined as laid out below. If you are unable to complete the assignment, you should still submit what you have, as you can get partial credit.

Total Possible Assignment Mark: 100% (10% of total course grade)

- Part 1: Arduino to Processing (40%)
- Part 2: Processing to Arduino (40%)
- Part 3: Multi-directional Communication (15%)

Marks will be given for completing the task, as well as the quality of the code you write. You should include comments which explain your code, and try and make your code easy to understand. Some of the marks awarded will be given for doing more than the minimum requirement, so if you have time you should try to add some extra flair to your designs.