# Group Project: **USB controller**

**Student Name:** James Poirier

**Date:** Friday, March 15, 2019 to Sunday March 24

## Work Scheduled for Week (march break)

* Get the Teensy Arduino (should be shipped on March break)
* Code the Teensy to be a controller

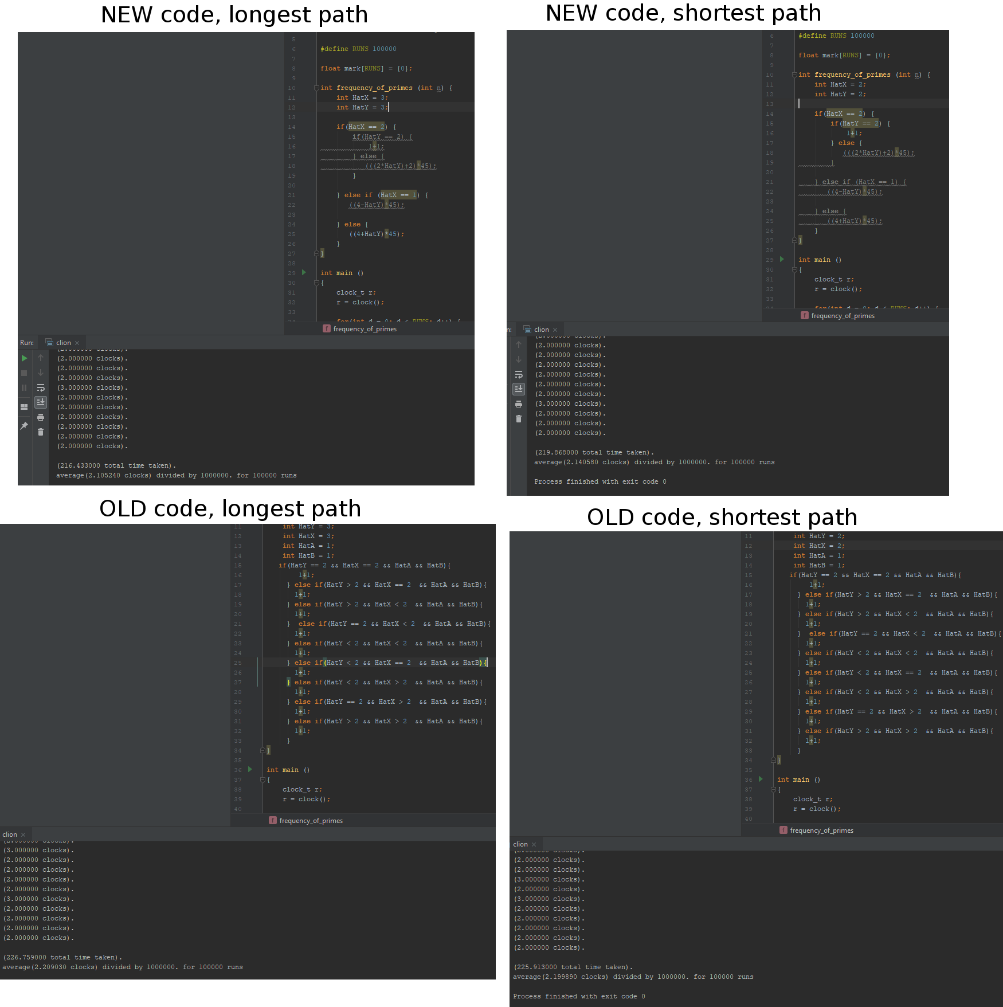
## Work Completed

* Over March break the Teensy came in.
* Helped Paul solder the circuit for the Mechanical key switches on the perf-board
* Recoded the Hat Switch on the Leonardo

## Work Comments and Results

* Previously the hat switch was a chain of Ifs checking which buttons where held and which where not to set the direction. I changed it to reduce the amount of checks.

The Left and Right buttons as well as the Up and Down buttons have a variable that can have the values 1,2,3. ‘1’ for one direction, ‘3’ for the other direction and ‘2’ for no direction.   
  
then to change the hat switch which has values -1 to 8 (but in 45 degree increments) I use a “if, else if, else” with the ‘X’ 123 value and set the hat switch’s state with a equation that uses the ‘Y’ 123 value.   
  
The ‘2’ values are the first to get checked because it is the most likely state.  
  
I created a program that can check ‘efficiency’, here are my results



(pic for proof but ill explain)

I made a program that gets the time, runs a function X times, then gets the time again minus the first time. This gives the difference in time in milliseconds (noted as clocks in pic). For these tests I ran the function 1000000 times, meaning if you divide clocks by 1000000 that’s about how long each run though of the code takes. I save that clock value to a list. I run the 1million times test 100000 times, this is to account for variance, I add up all 100000 clock times and divide to get an average.

My old code averages 2.2ms/1000000calls and my new code averages 2.1ms/1000000calls

Because I’m running each test of 1000000, 100000 times there is no difference in the average when I rerun the program where as when I was only running it 100 times it would be +/- 0.1ms between runs.

**Date:** Monday, March 25, 2019

## Work Scheduled

* Rewrite the Leonardo’s code for the teensy

## Work Completed

* Created a diagram showing what pins I’m mapping to what buttons.
* Created the basis for the controller, defining and pinMode(ing) pins
* Created and tests the analog joysticks (both left and right)

## Work Comments and Results

* Having a diagram that shows what buttons have what pins makes troubleshooting a lot easier
* I couldn’t do any work this day at school because of the software require, that’s why I did the diagram and defining. (analogs I did at home)

**Date:** Tuesday, March 26, 2019

## Work Scheduled

* Rewrite the Leonardo’s code for the teensy

## Work Completed

* Programmed 16 buttons
* Ported the Hat switch code from the LEO
* Updated the pinout because I had gotten my left and rights mixed up
* Went though each of the examples and <https://www.pjrc.com/teensy/td_joystick.html>.

## Work Comments and Results

Today I stayed home to get work done.

The “typical gamepad Layout” is not the same as xboxs layout. This is 2341 and xbox is 1243 (starting from the bottom)

**Date:** Wednesday, March 27, 2019

## Work Scheduled

* Trouble shoot every button and analog stick to ensure they work and it’s ready to go On the Teensy

## Work Completed

* Reordered some of the pins so they where closer to where the components would be (having L2 then L1 further buttons and on further pins)
* Installed programs to help with troubleshooting and games to test if they register the controller and the IDE
* Fixed a problem with the analog sticks
* Grabbed a work machine and set it up (did not work at first)

## Work Comments and Results

The Leonardo, for analog uses Xaxis, Yaxis, Rxaxis, Ryaxis, Zaxis and Rzaxis. x, y, Rx and Ry being the two sticks.

teensy has Xaxis, Yaxis, Zaxis, Zraxis, sliderL, and slider. X, y, Z, Zr being the two sticks

another problem is one of the analog sticks on the prototype is broken. It’s at default state around 300 rather then 512. This makes you constantly rotate in one direction. If I just do a +200 or add a resistor then that changes, it’s minimum value to 200 which means it never fully goes in that direction (which is also just as bad of a problem) I have not found a fix

when testing some games some use 2341 and some use 1243(see last day) I have not found a solution to this yet or why it happened.

**Date:** Thursday, March 28, 2019

## Work Scheduled

* Finish prototype for sprint

## Work Completed

* Wired Leonardo into the clay prototype
* Some buttons where not on the prototype and had to be removed from the code or they would be open circuits and potentially give wrong inputs
* Researched an alternate way to wire the buttons that will save on components (no pullup resistor)

## Work Comments and Results

* In my haste I plugged the dpads ground into 3.3v and because of that it was not fully functional.
* Brian didn’t like that I used pull-down resistors. I’ve done some research and learned that holding TTL inputs LOW uses more current. The chips the Teensy (k20 series) use do not define CMOS or TTL. I also haven’t seen any forum page call a specific one as better for an Arduino (because it’s cmos)
* This did cause me to look into finding a better way. On the Teeny I can make it so that a no connect is read as high. This will let me wire a button from pin to ground.