**CDEFO Week 7**

**Week 6-7:**

- **Core functionality needs to be done**
  - The Arduino should be to take a command from an NFC tag, and execute either on itself or on the Host Machine
  - C++ functions for media control, Bash scripting, and controlling digouts on Arduino must be working
  - A few modules should be made for demonstrative purposes
- Materials for user experiences should have all arrived
- At least 4 user experiences by this point
- Majority of C++ functions will have been written and tested on Arduino and Host Machine (continuous)
- Model/print enclosures (5 days)

Core functionality is done!!! My library is in a perfectly working state, and I have to say its pretty well optimized at this point. There were some issues involving String objects in Arduino and the fact that there was only 2kB of RAM, so the Arduino would consistently crash when changing scope (when it would switch scopes, it would need to destruct so many string objects that it would hang quite a noticeable amount and crash more often than not). However, after switching everything to character arrays (i.e. c_strings) I saved a lot of memory and the Arduino even sped up a noticeable amount.

Also, it occurred to me last week that a lot of the Arduino functions will need to run simultaneously, so I needed to make the driver multithreaded. The only problem with this, though, is that Arduinos only have 1 core, which means I can't multitask discretely. That normally wouldn't be an issue, *I can just implement threading using the stack OH WAIT…. Arduinos don't have stacks either ?!*

So… I needed to basically implement a really ghetto version of hyperthreading called protothreading, which has some documentation here. Thank god that a sizeable portion of processing is going to be offloaded to the Python machine! It's still staging right now, and I don't really want to merge it when everything is working as is right now, I might branch it like I did earlier. Speaking of which, while experimenting with the RAM limitations, I packed all of my static functions into an actual C++ class that can be instantiated. As expected, this did the opposite of free up memory, but it made the driver sketch much easier to look at, with much fewer function calls. This might be the final version that I recommend, and I'll provide the static version to people who want more control over the program flow without having to edit libraries. All I need to do is make it threadable and replace the String objects with character arrays.

Now that I have the core stuff down entirely, I can focus on making each function work together flawlessly by testing for edge cases, like when the transition between 2 lighting patterns is particularly jarring, I can program in a delay with a gradient. Also, I can add things like idle lighting animations and sounds that will make the installation seem much more interactive and responsive.

Now, while I do have module functions ready to run, I don't have their enclosures built yet, since there was some hangup related to funds and redesigning them in a more budget conscious manner. The "room" that the demonstration will be setup in will be surrounded on 3 sides by 10 yards of black diffusion-type curtain, and behind those three walls will be a bar of LEDs about a meter long that are

being diffused by a 7 ft. * 5ft. white nylon silk diffusion fabric. All of these will be supported by ¾" wooden dowel rods connected with 3D printed braces. Imagine the LED bars on the outer perimeter, shining through the white diffusion fabric, and then bleeding through to the viewer to make a understated mood lighting type look. The enclosures for the individual modules and breadboards will be laser cut through the Ponoko website I linked last week.

John and I's experiences are progressing well enough. I will try to find my third experience after Spring Break, and use Spring Break itself to receive the materials and actually build the experiences in full. The only thing I really need to figure out about John's experience is how I'm going to make the inner curtain ripple like water. I've also added a music visualizer using some left over LEDs for my own experience. So, just like every other week, I'm behind on the "User Experiences" and the reason for that hasn't changed, really. I spent a fair bit more time on the project this week than last week, around 15 hours.