21M.370 Digital Instrument Design
Lab assignment 2 - Feb 22 to Mar 1

Goals:

1. Install Python and be able to run a python script within an IDE of your choice
2. Be able to receive a control sequence from python and use it to control a PD patch.
3. Create your own sequence inside of python
4. Install Arduino and the ESP32 framework
5. Solder headers to the ESP32, 4 potentiometers, and 4 buttons
6. Be able to upload test code to the ESP32 and monitor in Arduino and in Python

---

Installing Python:

The first step is to install Python 3, along with the python-osc and pyserial modules. Look at the Python readme on the github for info:
- https://github.com/collaborative-music-lab/NIME/tree/master/Python
- be sure install python 3, pyserial, and python-osc
- if you use sublime, be sure to follow the faq on running python3 scripts from within sublime

---

Working with Python and PD:

For your synth and control PD patch you can use the example provided or make your own.
- you will need to follow the setting PD filepath tutorial video to use the PD patches in our repo

A couple of tips:

- you can use the 'properties' of your UI objects to add labels and configure how the objects look and function
- you can 'pd' objects to make subpatches, which work just like regular patches
- look in the 'pd sending-params-to-synth' subpatch to see how to format messages for sending to synth objects

For your python script, you are welcome to use the example provided to send sequences to 8steps. For this assignment it would be sufficient to just add more sequences to the sequence array. But you are welcome to go beyond this if you wish, and we will talk about more uses for Python in the future. Be sure to watch the video on PD and Python, which will show how to use Python to format OSC messages efficiently for use with automatonism.

The list of resources for this section of the lab:

1. Main class youtube video playlist
    1. https://youtube.com/playlist?list=PLHQPaeahzMrMU7iR_pKgxRs17RmJNpgPA
2. Video tutorials in that playlist that you might want to review this week:
    1. Setting the PD filepath to use PD patches in the class github repo
    2. working with message boxes in PD
    3. working with signals in modular synths
    4. working with automatonism and PD generally
    5. frequency and pitch representations
3. The GitHub Python readme gives information on how to install Python, and how to set up Sublime Text to use Python 3.
4. The demo instrument provided for this week's lab, located in NIME/PureData/Instruments/:

    1. ExtCtrlSubSynth is a subtractive synthesizer with a demo of a control patch saved within it: subSynth-ctrl.pd. This has all of the functionality needed to complete this week's lab. **You will need to open both the main.pd and the subSynth-ctrl.pd patches.**
    2. The 370_Lab2.py script will work with the above PD patches.
5. Also check the github's PD readme for more on working with PD.

Soldering the ESP32 and sensors:

Assemble your kit

The first thing we will do is assemble our kit. Assemble a cardboard box, and put into it:

1. An ESP32
2. A m370 breakout PCB
3. 4 potentiometers
4. 4 buttons
5. 2 sets of 40-pin socket-to-socket cables
6. micro-usb cable if you need it

Install software

To install Arduino and support for the ESP32 within arduino follow the instructions at https://github.com/collaborative-music-lab/NIME/blob/master/ESP32/README.md

Solder ESP32 and sensor

The following video tutorials cover basic soldering and electronics for our first instrument:

Main class youtube playlist
Soldering the ESP32

Assembling potentiometers
Assembling buttons
Plugging into the breakout board
Basic electronics

You will need to solder:
- the ESP32
- 4 potentiometers
- 4 buttons

Test in Arduino
Once everything is soldered, you can plug into the breakout board (watch the videos above if you need help) and test your sensors using code similar to the lab3_test firmware in NIME/Class/Labs/lab 3