Disambiguation:

This project has a lot of moving parts (Motion source, PadTest, Tesseract, eclipse, Audacity, Visual studio code, window switching, Quartus), which can be very overwhelming to deal with. Because of this the readme is broken into parts, external software, custom software, hardware, then implementation. Please note the phone can only be Android OS and the computer being used must be Windows OS. Nothing else will work.

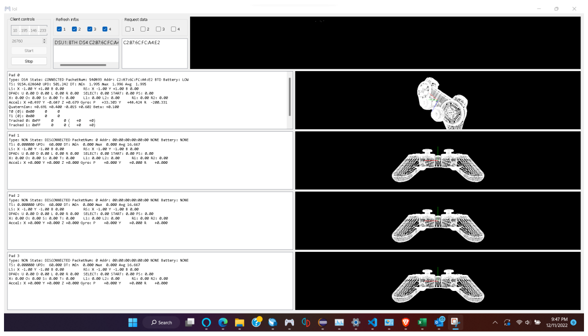
Credit to the code in this final project goes to ECE department for the lab 62 code, the anonymous creators of motion source and padtest, the entire team responsible for tesseract, and the python libraries I used, and Pramod and Ian the CAs who created small helper tools for wave table synthesis, and python JTAG connection respectively. Special thanks for help with debugging goes out to the TA’s Phil and Zayd, could not have done it without them. All the love and emotional support goes out to Pramod though.

External Software:

The first two steps are to set up the server that stores the motion control data
https://cemuhook.sshnuke.net/padudpserver.html
Basically the first step is to download the motion source app online using the tutorial link above onto an android phone then setting up the server configs on the phone using the motion source app (with your settings set to "Best" and "game").

https://files.sshnuke.net/PadTest_1011.zip
The next step is downloading the padtest program from the link above on the host computer.

And putting the same IP configuration from the phone and port onto the padtest program. If everything is done by this step, if you start the connection, you will be able to see the controller move on your screen. Also note the path to the application.



10.195.146.233
26760

Refresh infos: [✓ 1] [✓ 2] [✓ 3] [✓ 4]
Request data: ☐ 1 ☐ 2 ☐ 3 ☐ 4

DSU1: BTH DS4 C2:87:6C:FC:A4

C2:87:6C:FC:A4:E2

```
Pad 0
Type: DS4 State: CONNECTED PacketNum: 540893 Addr: C2:87:6C:FC:A4:E2 BTD Battery: LOW
TS: 9154.628640 UPD: 501.242 DT: Min  1.995 Max  1.996 Avg  1.995
LS: X -1.00 Y +1.00 B 0.00          RS: X -1.00 Y +1.00 B 0.00
DPAD: U 0.00 D 0.00 L 0.00 R 0.00  SELECT: 0.00 START: 0.00 PS: 0.00
X: 0.00 O: 0.00 S: 0.00 T: 0.00    L1: 0.00 L2: 0.00      R1: 0.00 R2: 0.00
Accel: X +0.497 Y -0.687 Z +0.679 Gyro: P   +33.303 Y   +48.424 R  -200.331
Quaternion: +0.691 +0.400 -0.015 +0.602 Beta: +0.100
T0 (0): 0x00     0     0
T1 (0): 0x00     0     0
Tracked 0: 0xFF     0     0 (   +0    +0)
Tracked 1: 0xFF     0     0 (   +0    +0)
```

```
Pad 1
Type: NON State: DISCONNECTED PacketNum: 0 Addr: 00:00:00:00:00:00 NONE Battery: NONE
TS: 0.000000 UPD:  60.000 DT: Min  0.000 Max  0.000 Avg 16.667
LS: X -1.00 Y -1.00 B 0.00          RS: X -1.00 Y -1.00 B 0.00
DPAD: U 0.00 D 0.00 L 0.00 R 0.00  SELECT: 0.00 START: 0.00 PS: 0.00
X: 0.00 O: 0.00 S: 0.00 T: 0.00    L1: 0.00 L2: 0.00      R1: 0.00 R2: 0.00
Accel: X +0.000 Y +0.000 Z +0.000 Gyro: P    +0.000 Y    +0.000 R    +0.000
```

```
Pad 2
Type: NON State: DISCONNECTED PacketNum: 0 Addr: 00:00:00:00:00:00 NONE Battery: NONE
TS: 0.000000 UPD:  60.000 DT: Min  0.000 Max  0.000 Avg 16.667
LS: X -1.00 Y -1.00 B 0.00          RS: X -1.00 Y -1.00 B 0.00
DPAD: U 0.00 D 0.00 L 0.00 R 0.00  SELECT: 0.00 START: 0.00 PS: 0.00
X: 0.00 O: 0.00 S: 0.00 T: 0.00    L1: 0.00 L2: 0.00      R1: 0.00 R2: 0.00
Accel: X +0.000 Y +0.000 Z +0.000 Gyro: P    +0.000 Y    +0.000 R    +0.000
```

```
Pad 3
Type: NON State: DISCONNECTED PacketNum: 0 Addr: 00:00:00:00:00:00 NONE Battery: NONE
TS: 0.000000 UPD:  60.000 DT: Min  0.000 Max  0.000 Avg 16.667
LS: X -1.00 Y -1.00 B 0.00          RS: X -1.00 Y -1.00 B 0.00
DPAD: U 0.00 D 0.00 L 0.00 R 0.00  SELECT: 0.00 START: 0.00 PS: 0.00
X: 0.00 O: 0.00 S: 0.00 T: 0.00    L1: 0.00 L2: 0.00      R1: 0.00 R2: 0.00
Accel: X +0.000 Y +0.000 Z +0.000 Gyro: P    +0.000 Y    +0.000 R    +0.000
```

Then downloading tesseract off of this link::
https://sourceforge.net/projects/tesseract-ocr.mirror/

After going through the setup wizard, make sure to record what your path for tesseract was. You will need it for the software step.

Other Software:

Luckily, I included a few tests to see whether or not the programs actually worked.

First you need to download all dependencies (ran out of time to grab them all but in python, they just read

```
import time
#import cv2
import mss
import numpy
import pytesseract
import re
import win32gui, win32con
import os
import subprocess
import jtag_uart
```

For the main script, “ece385finaldemo.py,” and

```
import os
import subprocess
import time
import win32gui
```

For the windowactivation.py helper script.

As mentioned in the prior section, please note that you have to change your path

```
current_dir =r'C:\Users\matte\Downloads\PadTest_1011'
```

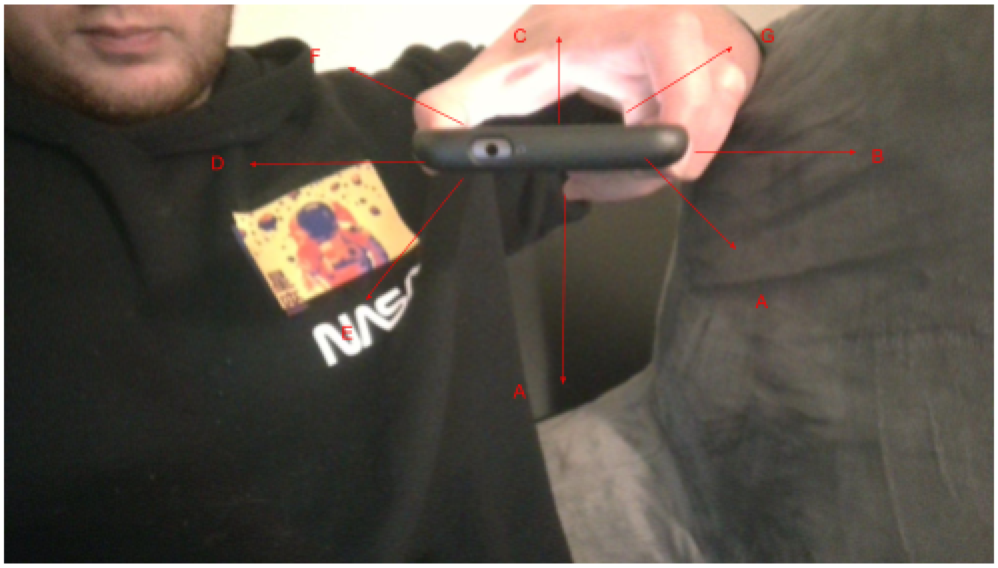
In lines 6 and 12 of the windowactivation.py script.

And you will also need to change line 68:

```
pytesseract.pytesseract.tesseract_cmd = r'C:\Program
Files\Tesseract-OCR\tesseract.exe'
```

To the proper path for tesseract.

If all of this works, you can run a prototype with two phones called gyroinputdemo.py after running the windowactivation.py script. It should be able to play notes out of your computer speaker. The notes playing are as follows.



Eclipse is fairly simple, all you do is run the same script that was provided for I2C on the course page and stop the connection with the nios ii once it is flashed onto the board.

System Verilog portions:

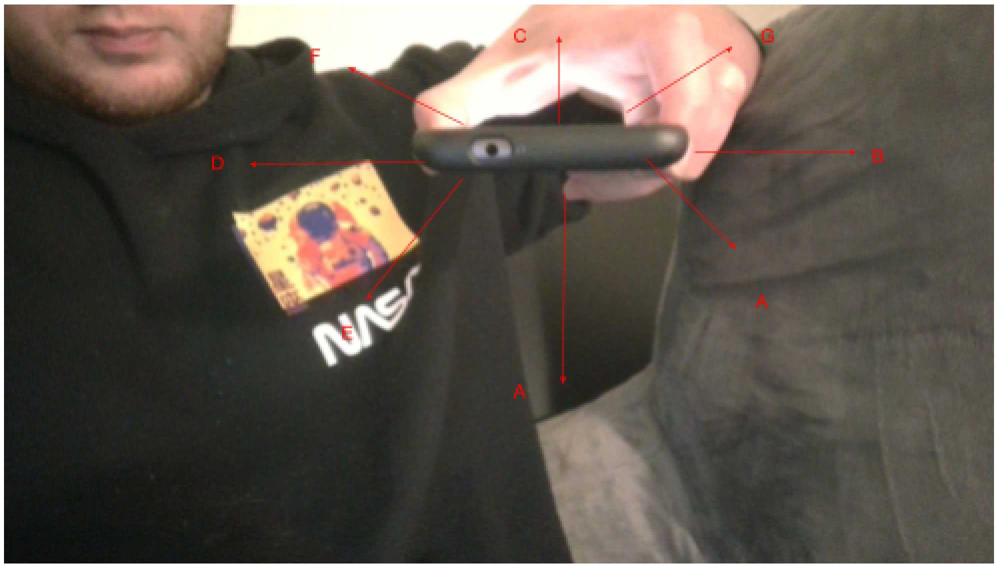
This part is simple. Just flash the project SOF onto the DE-10 lite board. Make sure the eighth switch on the board is facing upwards to get the data from JTAG to the board registers.

Running program.

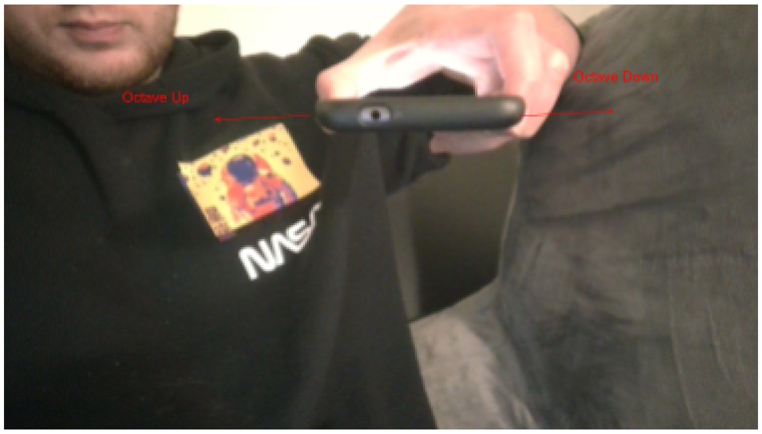
Order to go in for implementation while running is

1. Complete partial program setup
2. JTAG Programmer
3.Nios ii Eclipse Flash programming
4.End the JTAG connection with Eclipse
5. Run windowactivation.py
6. Run ece385finaldemo.py
7. Move the phones as you wish to play notes.

The note setups are as follows for the phone connected to the DSU controller tab



While the octave setups are as follows from the lol tab:



I designed it like this in order to be easy to play and sound good even if you swing it around, as well as to be able to have rest notes. The accelerometer data is prone to error even if it is completely still.

I made it this way in order to make it so that I can play every note in a key with reusing only one, while not making it impossible to play with little training. Adding sharps and flats in randomly with the control scheme would make it sound horrible.

If anything breaks it wasn't me.
If you want an explanation on how any of this works in further detail, message me on discord @mattsaidyeet#0152

Github link is
messes2/ECE-385-Final-project: I made a motion controlled digital synthesizer known as a puppeteer theremin that uses AI to get motion control data from a server hosted on a phone onto an FPGA. (github.com)