Davis 109006201
https://github.com/davizjc/Embedded-System-Lab-Hw3.git
周志偉

Homework 3 Serial Port and Audio Synthesis Report

1. SPI protocol
    a. Connect the pin like lab 7_1_SPI , for this homework pin d9 will be our digital out. Pin 9 - 12 will be connected to the SPIslave pin. I decided to use a for loop for each mode starting from mode 0 until mode 2. Firstly I initialize the value to be zero, so if mode equals zero, value is zero. If mode equal 1 then 1, mode equal 2, then value is 2. The master will send the mode 1 time and send the value 2 times. Then it will read the response from the slave. Cs will be set to zero and 1 to deselect and select the device.

```
Start.
mode = 0
Default reply to master: 0x00
Read from master: v = 0
recieve value = 0
Start.
mode = 1
Read from master: v = 0
recieve value = 1
Start.
mode = 2
Read from master: v = 0
recieve value = 2
```

2. I2C protocol
    a. Kinda the same as SPI but for I2C there were only Connect pins D15 to A5 and D14 to A4. There will only be two wires SDA and SCL. I use D14 and D15 for i2c_slave, A4 and A5 for mas(i2c_master). For each round each mode will be sent. For this one, the master will first send the mode. Then the Slave Read Addressed will send back a value and the mode, then the master will write the initial mode and value. The master has requested a read from this slave, so now depend on the mode the value will be the same , add one , or add two. In the end the master will read the mode. I did three rounds for three modes.

```
round 1 mode 0
mode send = 0
Slave Read Addressed: mode=0, value=1
master thread write: mode=0, value=1
Slave Write: mode=0, value=1
master thread read: mode=0, value=1
round 2 mode 1
mode send = 1
Slave Read Addressed: mode=1, value=1
master thread write: mode=1, value=1
Slave Write: mode=1, value=2
master thread read: mode=1, value=2
round 3 mode 2
mode send = 2
Slave Read Addressed: mode=2, value=1
master thread write: mode=2, value=1
Slave Write: mode=2, value=3
master thread read: mode=2, value=3
```

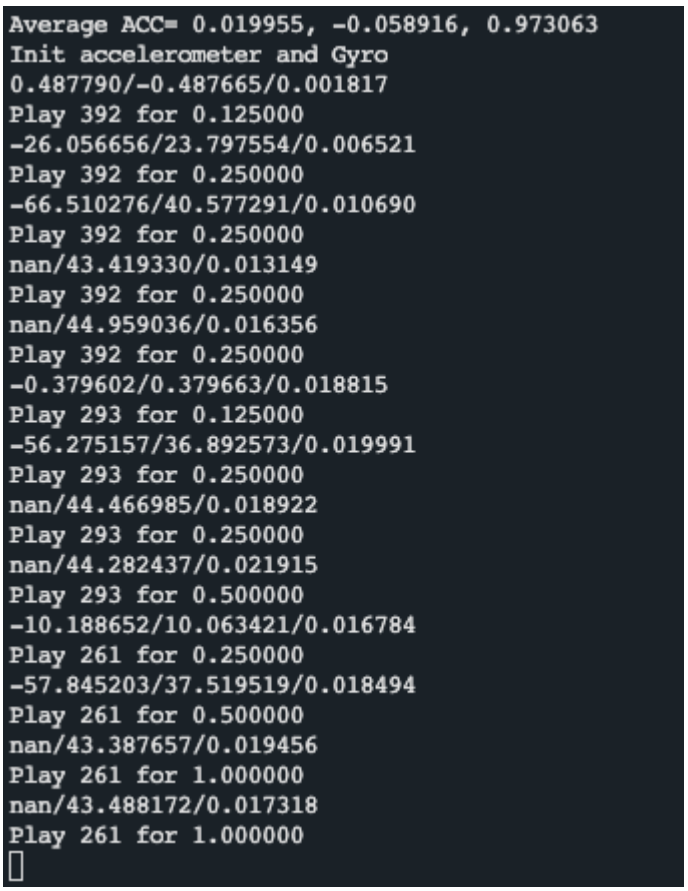
3. UART
    a. Connect pins D0 to D10 and D1 to D9. For our global variable will we use static BufferedSerial device1(D10, D9) and static BufferedSerial device2(D1, D0); .Basically two device, device 1 for master and device 2 for the slave. For this one i just put 6 number in a array (mode , value ,mode1 , value , mode2, value) kinda the same as the previous one and from the master I write to the slave. The slave will then read the mode and value that was send and write back. In the main set desired properties (9600-8-N-1) for each device.

```
start
Slave Read: mode=0, value=0
Slave Write: mode=0, value=0
Slave Read: mode=1, value=0
Slave Write: mode=1, value=1
Slave Read: mode=2, value=0
Slave Write: mode=2, value=2
```

4. capture and average 10 samples of accelerometer and gyro
    a. Inside the gyro and accelerometer from 8_4_Acc_Gyro instead of using 200 samples just change it to 10.

```
Average ACC= -0.018055, -0.073170, 0.966411
Average ACC= 0.043712, -0.093125, 1.034830
Average ACC= -0.052264, -0.277475, 0.899893
Average ACC= -0.026607, -0.306933, 0.988267
Average ACC= 0.003801, -0.311684, 0.916048
Average ACC= 0.002851, -0.280326, 0.894192
Average ACC= -0.006652, -0.251818, 0.937903
Average ACC= -0.001901, -0.240415, 0.944555
```

5. Please modify the MBED program in 3.2 (for keyboards and interrupt) in MBED lab 8 such that we can use the previously calculated pitch, roll and yaw from averaged samples to determine the length of a playing note. The length of a note will only be in 1/8, 1/4, 1/2, and 1 second.
    a. Set up the MBED board and circuit board like lab 8_2_Synthesizer, and connect it to the speaker. Remember to add the library from lab 8_4_Acc_Gyro so accelerometer.h and gyro will be able to be used. In this homework we generate pitch,roll and yaw. When the MBED board moves it will change these three numbers. I wrote that when pitch >10 or roll > 10 then the index pitch will increase.And since we only have 4 note lengths if(index_pitch>3) index_pitch=3, else index_pitch--. To make the three buttons play different notes I use a not really efficient way. The noteLength will be changed for each button but the note play will be the same. For example when keyboard0.fall(queue.event(startRecord)); inside record function “ idNote[indexNote]=queue_note.call(playNote, song3[index_pitch], noteLength[index_pitch] “ will be called and inside song3 i have G_4 sound play.



Keyboard 2 was pressed, the note G will be play and as I tilted the length of each note play will be longer from ⅛ until 1. When I press keyboard 1 it is the same but for D, and keyboard 0 for C.