Friday, March 16

I set up my digital portfolio page and inserted my design specs onto the newly created "Personal Project Semester 2" page. I started to reread the Adafruit Bluefruit Feather nRF52 page to familiarize myself with the terms again, and I realized how much of this information I forgot after finishing the project! I did, however, learn a new term: bricking. If you brick a piece of hardware, then you mess it up and prevent it from working properly. I created my materials list and inserted it into my digital portfolio.

Monday, March 19

I picked up where I left off at the end of first semester and had my phone communicate with my laptop over Bluetooth, with the help of the Adafruit Bluefruit App. I used the "bleuart_cmdmode" example code that is in the "Adafruit_BluefruitLE_nRF51" library and sent messages back and forth between my laptop and phone. Once I did this, however, the port stopped appearing and my laptop could not upload files to the Feather. The only available port was the actual Bluetooth port, and I don't know why nothing showed up.

Tuesday, March 20

I edited my digital portfolio and made sure that everything is up to date. Then, I tried to do the same thing as yesterday (using the bleuart_cmdmode code), but the port still didn't appear in Arduino. I replaced the USB to USB Micro cord and used different USB ports on my laptop, but the Feather port still didn't show up. It was very strange because the code worked yesterday and suddenly it didn't work. So, I tried to open it and run the program with a school computer in Arduino.

Wednesday, March 21

I undid all of the sewing that I did in Semester 1 because the Feather wasn't working. Every time I plugged it into the computer (and I tried my personal laptop as well as the school computer), it didn't even show up as a port option. I even restarted my laptop to try to get the Feather to communicate with my Mac. I tried a different Feather that I got from the FabLab, and it worked! It wasn't the USB port, the cable, or anything else. It was the Feather itself.

Tuesday, April 3

Now that I'm using a different Feather, I noticed that some of the pins that I need (like GND and SCL) had solder in them. I would have used the desoldering gun like I did last semester when I removed solder from the Flora, but we couldn't find it. So, Mr. Rudolf taught me how to use a copper braid and flux to remove the solder. I had a bit of trouble with it at first, but I soon got the hang of it and removed all the excess solder from the pins. I'm slightly concerned that, since I was so bad at removing the solder at first, I burned the pin too much that it cannot conduct anymore. See the workflow for steps on how I removed all of the solder.

Wednesday, April 4

I had nothing to worry about; the pins worked even though a lot of it is burned because of the desoldering process. At the beginning of class, the Feather port wasn't showing up, so I started changing out Micro USB cords. Eventually, I found one that worked. Then, I moved code between the uart-bluetooth-cmd mode and the sensor code. I no longer get error messages, but it isn't sending the data that I want it to.

Thursday, April 5

I did some troubleshooting with the sensor at the beginning of class. Later, I continued moving ble code into the sensorapi code. Just to make sure, I tested that the ble code works with 9600 baud. Originally, the example code included 115200 baud. There is a bluetooth configuration tab in the ble code, so I thought that it might be necessary for the bluetooth to work. I copied and pasted it into another tab of my code, but nothing really happened.

Added BluefruitConfig.h from uartcmdmode example sketch
if (getUserInput(inputs, BUFSIZE)) ← example code
So i made if (lsm.getEvent(&accel, &mag, &gyro, &temp))
What is sensors_event_t accel, mag, gyro, temp; ?
Did research on it
Maybe this project isn't really possible: the uart one works bec the bluefruit is **connected** to my laptop → what happens if it isn't connected to the laptop? It seems like the program cannot start (factory reset, setup in general) unless the serial monitor is open on the computer. Will this work?

Friday, April 6

I worked with Mr. Rudolf on the code and showed him what I had done so far. With him, we tried the other Feather modules, but none of the other ones worked. He had me consider Logging, the speed differences, and if I wanted to store data. I told him that I just wanted to send all of the raw data to my iPhone, so I won't need any method to store data. While I was playing around with my code, I started wondering if this project was actually possible. When you open the serial monitor, this initializes the code.

Monday, April 9

I continued playing around with the code, and it seems like the code doesn't work without factory reset. In turn, the factory reset doesn't occur without opening serial monitor. So, this, again, might not actually work. I kept on going through the code and read https://www.arduino.cc/en/Hacking/LibraryTutorial and tried to figure out what #ifndef and #endif mean… They just define something and are helpful when using libraries. I also found out that None of the code works if you get rid of this bit:

```
#ifndef ESP8266
 while (!Serial);     // will pause Zero, Leonardo, etc until serial console opens
#endif
```

This code makes sure that the serial monitor is open before starting the code. I tried to remove it from the sensorapi code, and it still worked without this bit of code. Sensorapi4_4 (my edited version of sensorapi) doesn't do anything if you get rid of that part. It seems like I need the factory reset.

Tuesday, April 10

I wanted to change my code so that it would only start logging data once I opened the mobile app (Bluefruit LE App), and I actually got it to work! Initially, I thought that this piece of code:

```
while (! ble.isConnected()) {
      delay(500);
```

would stop the Feather from waiting for the serial monitor to open, but I later realized that it wasn't this piece of code. Anyway, I removed it, and the:

```
if(!lsm.begin())
```

command didn't work. I undid what I had just done. Again, I tried to get rid of the:

```
#ifndef ESP8266
 while (!Serial);     // will pause Zero, Leonardo, etc until serial console opens
#endif
```

code. I changed some things around and got rid of this piece of code on my Sensorapi4_4, and now it doesn't start logging data until I open the phone app. For example, I had commented out all of the serial.print commands to print the data on my computer. I undid just a few of these. However, the data is only being logged on my computer, and still not on my iPhone. Then, I tried to get rid of the factory reset at the beginning of my code. This caused this bit

```
ble.verbose(false);
```

to have an error and claimed that "ble" does not name a type. So, I just left the factory reset code in. While I was scrolling through, I noticed that this code:

```
 bool getUserInput(char buffer[], uint8_t maxSize)
{
  // timeout in 100 milliseconds
  TimeoutTimer timeout(100);

  memset(buffer, 0, maxSize);
  while( (!Serial.available()) && !timeout.expired() ) { delay(1); }

  if ( timeout.expired() ) return false;
  delay(2);

  uint8_t count=0;
```

```
  do
{
   count += Serial.readBytes(buffer+count, maxSize);
   delay(2);
  } while( (count < maxSize) && (Serial.available()) );

  return true;
}
```

Once I added that, I opened the serial monitor, and this message was just repeating:

```
<- 0

<- OK
AT+GAPGETCONN
```

I got rid of the above code to undo this problem.

if (lsm.getEvent(&accel, &mag, &gyro, &temp, BUFSIZE))

The other code has BUFSIZE→ what does that mean?

Wednesday, April 11

All of class, I kept on getting the error that the sensor wasn't detected and that I should check my wiring. So, I worked on my digital portfolio and put it / the code I'm referring to in clearer terms. I made the code green and the error messages red. Over the weekend or on Monday, I will sew the Feather and sensor to the vest so I don't have to deal with these alligator clips and subsequent error messages. Also, today, Mrs. Coble reminded us to document our process as we went along. But my process has all been over code, so what do I have to document? Once I get this running, I can document me using it and trying to figure out good ranges for the data, but for now, I don't really have anything to document.

Monday, April 16

I refurbished some of the other Feathers by desoldering them. I mainly just worked on one of the Feathers, but it had wires within the solder that I struggled with. I wasn't able to fully fix and remove all of the solder since the GND pin still has some wires (?) stuck inside of it. Tomorrow, I will sew my vest. I forgot the actual vest today, even though I actually remembered to bring the needles today.

Tuesday, April 17

I started to sew today (after forgetting my needle and thread for 3 days straight), but I spent most of my time trying to figure out how I would line up my "paths" for sewing. I saw that I would have to intersect my paths due to the location of the pinouts, so I figured out that I would use a piece of fabric underneath the fabric of the vest to let it not intersect. Refer to the diagrams.

Wednesday, April 18

I sewed some of the pinouts, but then I had to undo most of it because I accidentally positioned the boards incorrectly. With the way I had positioned it, the SND and SCL pins were facing the complete wrong way. This was super frustrating, but I'll resew these tomorrow, this time with the 9DOF facing the correct way.

Thursday, April 19

As I described yesterday, I resewed the 9DOF because I positioned it incorrectly. I had to sew one of the "paths" twice because the conductive thread actually snapped when I was sewing the first time. Since the thread is kept on bobbins, it is very coiled and is prone to tying in knots. Also, since the conductive thread is thicker and more abrasive, it sticks to itself more easily, furthering its ability to tie itself into knots. This characteristic of conductive thread makes it quite difficult to sew with in a hurry, so I've been taking my time with sewing the conductive thread.

Friday, April 20

Today, I made sure that the paths weren't intersecting by running the "Sensorapi" code. I sewed the last path, the one that required going underneath the actual vest layer. It took a bit of the class period to figure out how it would work. At first, I just had an extra layer underneath. Then, I realized that the conductive thread may still touch each other and stop the transferring of data. I redid the extra piece of fabric and made it double the length it's supposed to be so that I can fold it back over and cover it up. None of the threads seem to be touching, so it seems to have worked.

Monday, April 23

When it worked like it's supposed to, I started to sew the "covering" layer of extra fabric. This actually takes quite awhile, because I am also trying to avoid the thread from coming out on the top of the fabric. So, I'm putting the needle into the vest fabric shallowly for each stitch. Looking back on the speed of my sewing, I probably would've sewn faster if I wasn't so careful with the appearance of the vest.

Tuesday, April 24

Today, I finished up the sewing of the covering layer. After figuring out the correct positioning of the battery, I realized that I didn't position the Feather correctly. Since I didn't want to undo

several days of work, I cut a hole in the vest and hemmed the edges so that the battery can plug into the Feather. Last semester, I just let the battery wire go around the outside of the arm hole in the vest. However, when I sewed in the Feather, I rotated it so that the battery port faced the inside. This hole fixes the issue, but it created a lot more work for me. If I positioned it correctly the first time, I wouldn't even have had to sew any more.

Wednesday, April 25

I tried to look at other codes in the example library, but the coding for bluetooth seemed really different in each example code. In the "heartratemonitor" example code, I didn't recognize a lot of the code. First of all, the data it transmits is in Binary (a way that codes letters into short 4 digit numbers). Also, it "sends" the data to some other place that isn't the Feather, so I don't really think this code will be helpful for me. As for my actual code, I didn't change any of it today.

Thursday, April 26

I did research on what "BUFSIZE" means, since it's in the "bleuartcmdmode" code. BUFSIZE refers to the "buffer size" of a code. I don't really understand what that means, but it has to do with the transfer of information. I was also wondering what

println(F(---))

meant. I, again, don't fully understand this either, but it puts the text into flash memory. This somehow conserves space in the Feather and stores data in a different place. Here's the explanation that was posted[1]:

When you compile your program it says how much program memory (stored in flash) you are using and how much dynamic ram you are using.

If you use F() you can move constant strings to the program memory instead of the ram. This will take up space that will decrease the amount of other code you can write. But it will free up dynamic ram.

My chip has 32KB of Flash memory (2Kb taken up by bootloader) and 2KB or RAM. Sometimes I need to store lots of data in RAM so I move the constant strings to Flash memory. Sometimes my program has a lot of steps and I have a shortage of Flash memory so I leave the strings in RAM.

Monday, April 30

I moved documentation into my digital portfolio, including what I have of the code so far. I realized how behind I was on uploading my documentation. I've been documenting stuff, but I haven't put any of it into the digital portfolio. I created a Google Slide to hold all of my documentation materials, including some of the time lapses and videos that I've recorded.

Tuesday, May 1

[1] https://forum.arduino.cc/index.php?topic=428015.0

I was trying to think of the best ways to explain why I sewed the vest in the way that I did, so I created charts and diagrams of my sewing. I used a lot of layers of fabric, and the diagrams will help to explain why I used all these layers. This took a lot more time than I expected because, as I discussed yesterday, my documentation materials were everywhere, so I couldn't find the pictures I wanted very easily. Then, I had to upload things into Notability, draw on the picture, and then put it into the Google Slide. Tomorrow, I'll put in text to explain my process.

Wednesday, May 2

I continued my work with the digital portfolio and put much of it into a Google slide. Today, I put in textboxes to explain what I was doing and why I was doing it. I also sped up all my videos that I took originally (since they were 15 minutes long), edited them, and then inserted them into the slides.

Thursday, May 3

Using the orientation that I figured out last Tuesday (April 24), I started sewing the battery pack. Sewing this layer was actually considerably more difficult than sewing the other layers because I had already sewn so many layers. Throughout the process of sewing this layer, I continually had to make sure that the battery would stay snugly inside the pocket. By the end of class, I had sewn 3 edges of the pocket. Next class, I will finalize it and close up the pocket.

Monday, May 7

I started to look at the market for this vest, and I realized that if I go to the market with this, I can probably charge a lot for it. Skaters spend a lot on skating equipment (online, I saw that skaters spend between $35,000 and $50,000 a year[2]). I'm trying to figure out the market for skaters/skating paraphernalia. I also discovered that, since I accidentally left the LiPoly battery plugged in all weekend, it ran out of juice; I'll recharge it later. Mr. Dubick said that the FabLab has a charger for it, even though I've never seen one. Since I'll have to recharge this battery, I will have to sew it into the vest at a later time.

Tuesday, May 8

I went back and edited my daily journal entries. Sometimes, especially when I didn't stop working at the end of the day and just put bullet points in for what I did. So, today, I wrote what I did in full sentences while using my notes on what I did. Also, Mr. Dubick said that these experts will be coming in on May 22. I was hoping that they would help me out with my code, but that's 2 weeks away. I don't think I'll be able to figure out my code without them, but that's really far off.

[2] http://time.com/money/5136679/olympic-figure-skating-costs/

Wednesday, May 9

I looked at the Adafruit Forums (before, I was looking at the Arduino forums) and it seems like there is an example that sends information from a sensor to the nRF52. However, when I run that code, it doesn't work. It says that there's an error compiling for the board, even though I've been using code from the same library and what I've used so far has been working. After I looked at the code for awhile, I also realized that it uses a different sensor (the BNO055) and not the LSM9DS0 (what I'm using). I changed all the code that had the BNO055 and that mentioned the "bno," but that still doesn't fix my problem that the code doesn't compile for the Bluefruit Feather.

Thursday, May 10

I tried to download the nRF51 library, but Arduino keeps on saying that the zip file I downloaded from Github isn't a "valid" library. I don't know why it does this, as this site seems to be the unofficial official nRF51 library. I am actually using the nRF51 Feather, but the nRF52 library has been working for me up until now. Now, it suddenly doesn't compile for my board, and I don't know why.

Friday, May 11

I tried to fix the error ("Error compiling for board Feather") and started playing around with the code. I tried to understand all of the code that the example code has, but it's very difficult. Based on what I saw in the forum, however, this code seems to actually do what I want it to do. I'm going to keep trying with this code.

Monday, May 14

A comparison chart that I made today to help me figure out why I keep getting strange error messages:

BNO code	Same	Sensorapi
Same software on the Bluefruit LE	Adafruit_BluefruitLE_SPI ble(BLUEFRUIT_SPI_CS, BLUEFRUIT_SPI_IRQ, BLUEFRUIT_SPI_RST);	
These define values were the same after I changed the sensor (I changed it from the bno one to LSM9DS0)	#define LSM9DS0_XM_CS 10 #define LSM9DS0_GYRO_CS 9 #define LSM9DS0_SCLK 13	

	#define LSM9DS0_MISO 12 #define LSM9DS0_MOSI 11	
Has void displaySensorStatus		
Has void displayCalStatus		
Has void transmitCalStatus		
Has void initSensor - Includes displaysensordetails()		
		Has void configureSensor

Though this chart really helped me figure out what was going on, I keep getting this message:
no matching function for call to 'Adafruit_LSM9DS0::getSensor(sensor_t*)'
I copied this bit of code:

```
sensor_t accel, mavg, gyro, temp;

  lsm.getSensor(&accel, &mag, &gyro, &temp);
```

straight from the sensorapi code, but for some reason, it gets an error message when it's in the bnomodified code.

Thursday, May 18

Today, I helped Vivian engrave her shoe rack on the laser cutter.

Tuesday, May 22

Mr. Proffitt came in today, and he really helped me with my code. He inserted the sprintf function into my code, but the sprintf function does not support numbers in its float. So, we had to make another buffer to hold all of the information. I wouldn't have been able to complete this code and get the vest to work (it actually works now!) without Mr. Proffitt.