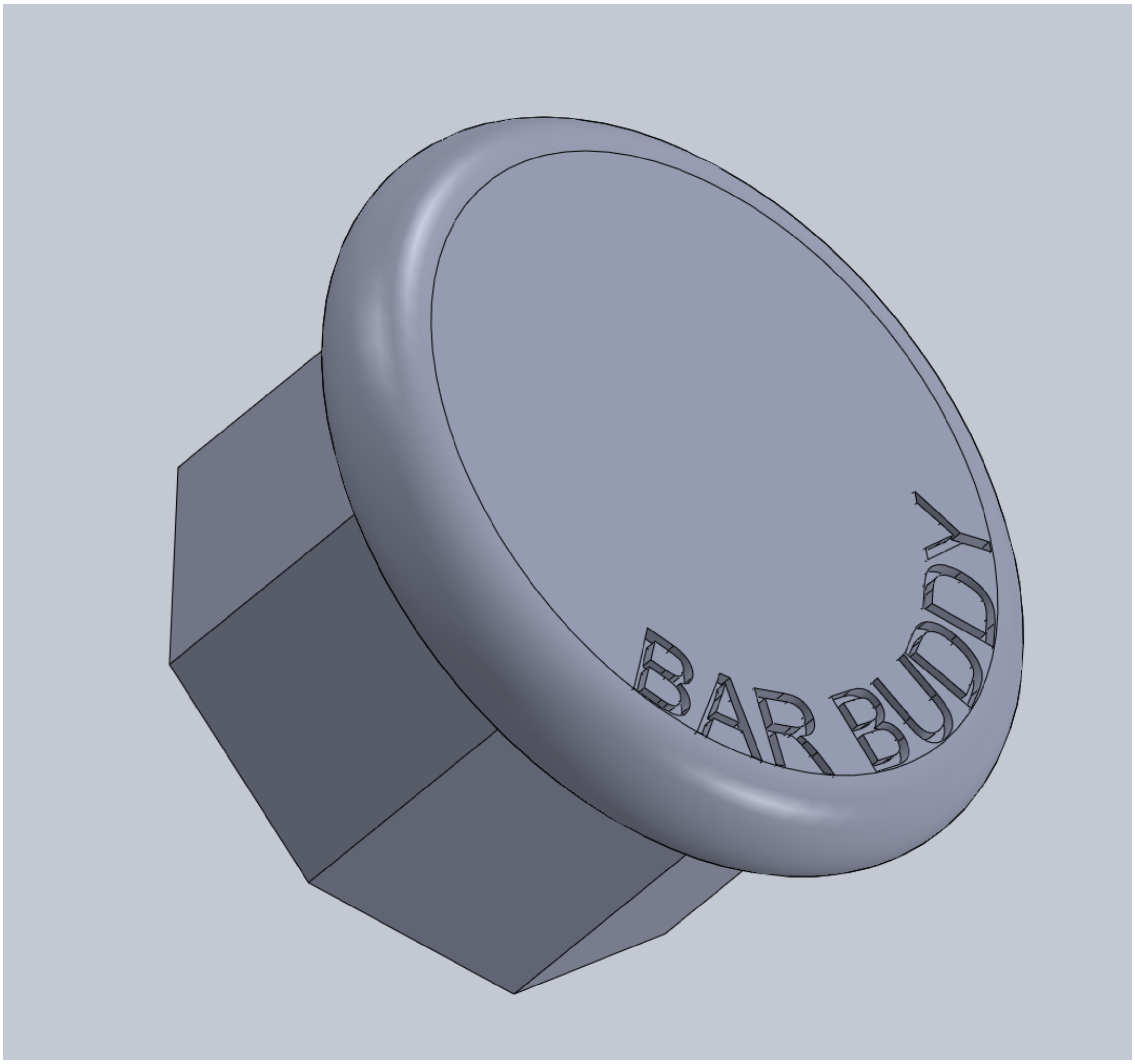
THE BAR BUDDY

CLIFF BRAUN



Date: December, 13, 2016.

Contents

1. Introduction

The Bar Buddy provides a motion analytics system to the weightlifter. There are two potential groups of customers for the product, that entail very different software implementation and interface, but very similar hardware requirements. The two groups are those seeking to automatically track their workouts, and more serious athletes looking for motion analytics to improve their form. Some hardware exists already to fulfill this function, but requires a physical tether from barbell to floor unit (`http://www.roguefitness.com/open-barbell-v2`).

1.0.1. *Automated Workout Tracking.* The largest group of potential customers is normal gym goers, who are looking for an easier way to track their workouts. Counting to ten becomes a challenge when working very hard. Beyond basic tracking functionality this market could be expanded by a prescribed workout app. The customer could perform listed exercises and watch as they are counted on screen, potentially with helpful feedback.

1.0.2. *Advanced Motion Analytics.* A smaller, but perhaps more interesting from an engineering standpoint, application is that of advanced motion analytics. This application would allow an athlete to receive real-time input on form. Potential examples of this include comparing workouts week over week for speed and stability, correcting poor form, and even suggesting auxiliary exercises to strengthen detected deficiencies.

2. Specifications

System specifications are shown in Table 1

Table 1. System Requirements

System	Requirement
Maximum Acceleration	10g
Minimum activity spacing	0.1s
Battery life	8h
Environmental Monitoring	Temp & Humidity
User interface	Wireless (Ideally BTLE)

3. Design Development

3.1. **WiPy Hardware Design.**

Initial hardware design was performed with the intent to use the "WiPy" microcontroller. This micro was chosen because of its support for MicroPython, built in WiFi, and meeting all other design requirements. The schematic created for this board is shown in Section A. The board consists of three major subsystems: power management, sensor support, and debug.

3.1.1. *Power Management.* Power for the system is provided from a standard 3.7V lithium polymer battery or the USB bus. The board contains a LT-1129 voltage regulator to convert battery voltages to the system voltage of 3.3V. A self-contained MCP73831T lithium polymer is also on board. Both the battery voltage and charger state are routed through voltage dividers to analogue input pins to allow system monitoring.

3.1.2. *Sensors.* The primary purpose of the board is to interface I2C motion and environment sensors to the micro controller to allow for data collection. The sensors utilized are the Bosch BNO055 integrated motion unit, and a Silicon Labs 7021 temperature and relative humidity sensor. These sensors share an I2C bus and associated hardware. The board provides mounting for an external crystal and associated components to increase accuracy of the BNO. An odd quirk of routing constraints: the BNO's I2C_ADDR_SEL signal is routed through a WiPy pin to allow for knitting of ground planes.

3.1.3. *Debug.* A CP2102 USB to UART bridge was included in an effort to ease system debugging as a micro-USB power was already included in the board for charging purposes. This component is relatively simple in use, but slightly complicates routing. USB signals should be routed on differential pairs, and incorporating ESD suppression diodes in the routing posed a challenge.

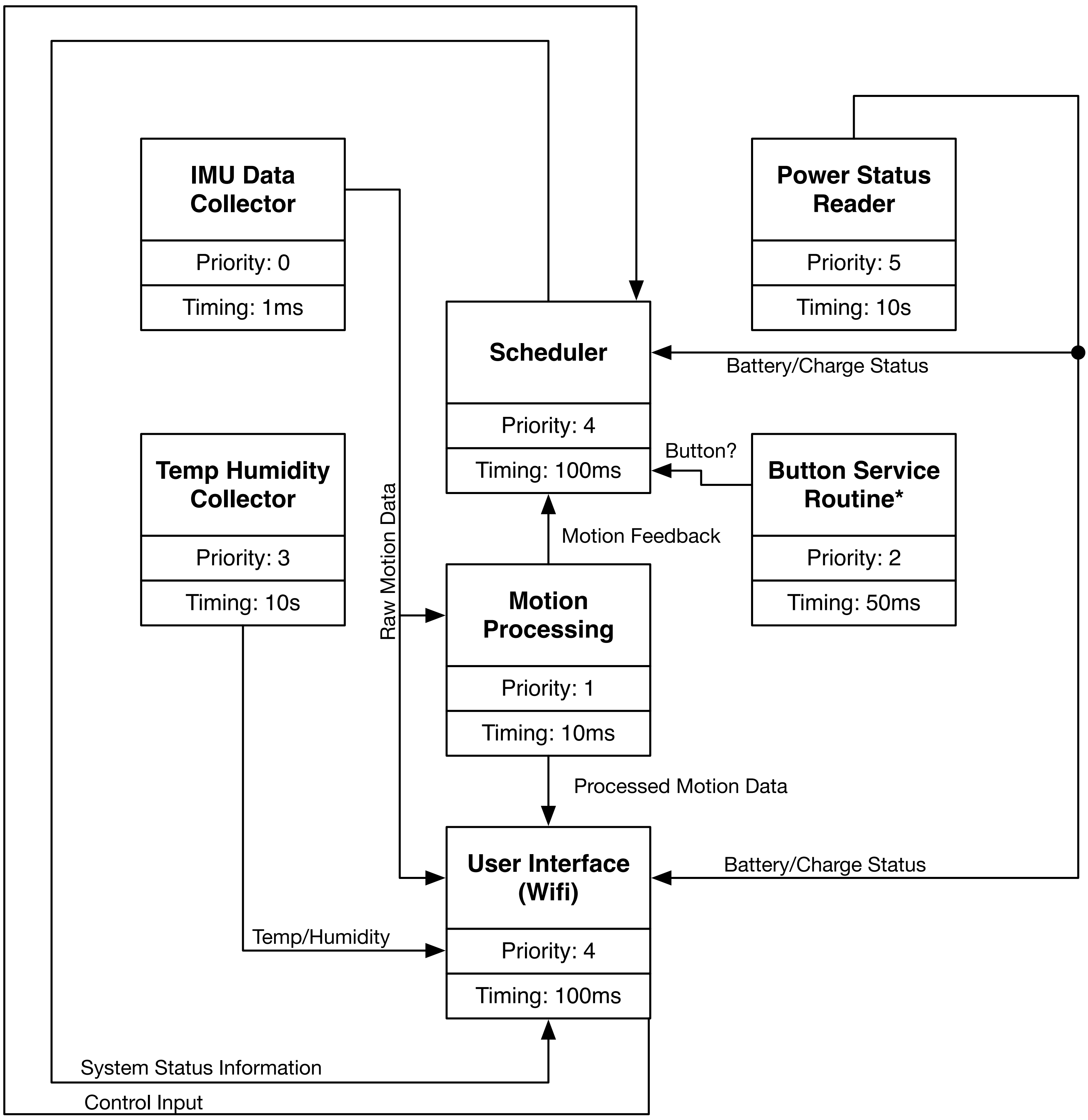
3.2. **BeagleBone Hardware "Design".**

During hardware bring-up the WiPy host board was put into an unworkable state, necessitating a redesign to allow for collection of data. The redesign was of greatly reduced scope, mainly aiming to validate that the Bosch sensor was capable of providing useful data for the product concept. The Beaglebone Black was chosen as the host for practical reasons: one was in the bin-o'-micros, and the Raspberry Pi suffers from a hardware I2C clock-stretching bug that prevents validation of the Bosch IMU connection. This hardware consisted of only three major components: a Beaglebone Black, a Bosch BNO055 breakout board from Adafruit, and a USB battery intended for phone use. This actually roughly imitates the intended design with a far more complicated host, several more regulators, and quite a bit more mass.

3.3. **Software Design.** Software design for the Bar Buddy utilizing the WiPy is described in the following pages.

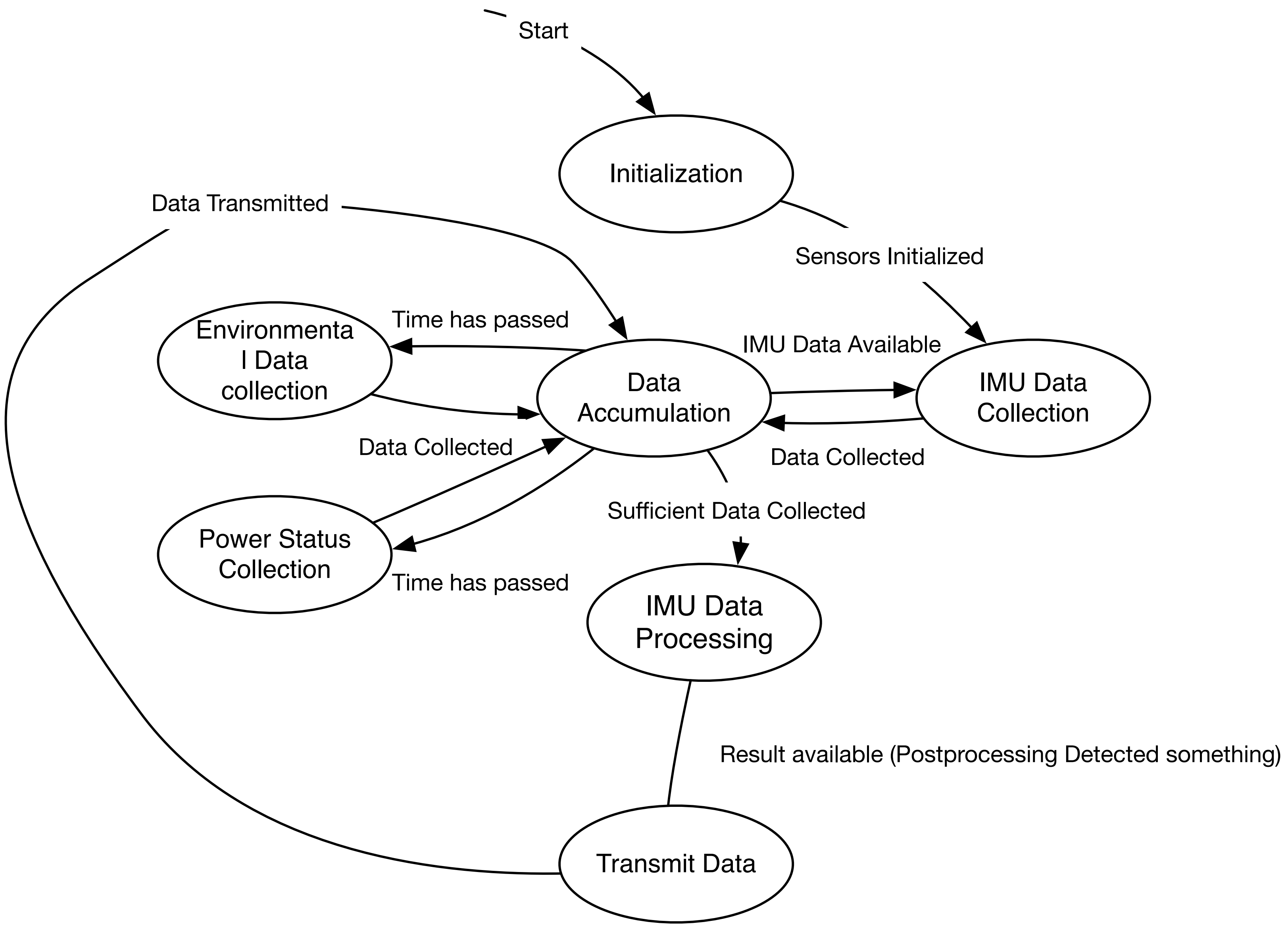
Bar Buddy Task Diagram
Cliff Braun
27-11-16

Notes:
Priority 0 is Highest
*Current hardware does not include button



Bar Buddy Scheduler States
Cliff Braun
27-11-16

Notes:
Priority 0 is Highest
*Current hardware does not include button



3.4. **Proof of Concept Software.** The Beaglebone proof of concept software consists mostly of a rough version of the "IMU Data Collection" routine, along with a utility to plot the resulting CSV files.

4. Results

4.1. **Hardware.** Hardware validation was performed on the major subsystems before the untimely death of the WiPy host. The eagle rendering, and actual board are shown in Figure 1

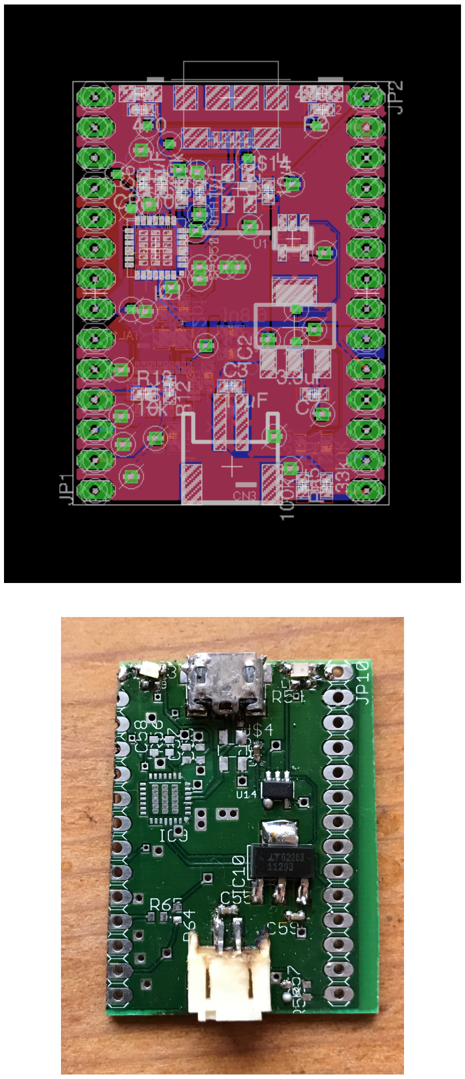


Figure 1. Top of board: rendered and actual.

4.1.1. *Power.* The power subsystem was nearly completely validated. The regulator comes up and provides stable voltages under system loads, but was not tested under loading corner conditions. The LiPo charger successfully charged the system battery, and appropriately actuated the charge signal LEDs. The two analogue inputs of charge state and battery level functioned as intended in the schematic, however an error was discovered in the design: the voltage divider multiplies by 3/4 rather than 1/4. Fortunately the fix for this is as simple as swapping the two resistors.

The major piece of missing validation of the power subsystem is a near-capacity stress test, it is unclear how the device will perform when drawing maximum current, or when attempting to run from a poor power supply. Additionally it is unclear how the system will function when the system is exclusively on USB power without a battery.

4.1.2. *Sensors.* Due to time constraints full validation of the sensors was not performed.

4.1.3. *Debug.* The CP2102 debug interface was, ironically, the source of most debugging for this effort. The chip is properly connected to the D+ and D- lines of the USB bus, and internal power regulators function properly, however the device will not enumerate on the USB bus of a host computer. Further debug will be performed using a breakout board from Sparkfun as a reference.

4.2. **Data Collection.** Data was collected on one occasion through a bench-press workout in several segments. Overall initial results were very positive, with little processing it is possible to pick out key features. It seems likely that the standard filtering and processing methods applied to activity tracker data will work for the Bar Buddy.

4.2.1. *Warm Up.* The initial recording made with the Beaglebone based Bar Buddy prototype was of a warm-up set of ten repetitions, with only the weight of the barbell (45lbs). Slightly filtered data is shown in Figure 2. The peak-to-peak accelerations in this plot are significantly higher than those in later plots.

4.2.2. *Short Sets.* The next segment of data collected was during ten sets of three repetitions each, performed every 30 seconds. The slightly filtered data is show in Figure 3. This plot shows repetitions not quite as cleanly as the previous one, as they were a bit heavier and involved slower acceleration. Two notable features of this plot are the ease of picking out sets of repetitions, and that the acceleration amplitudes decrease slightly as the lifter tires. Figure 4 shows a similar set of exercises, but with five sets of two repetitions. Loading was one again increased, and a corresponding decrease in peak acceleration can be seen. Additionally the last set seems to be cut off somewhat. There was a hard landing, and I believe this may have been a system reset.

4.2.3. *Paused Squats.* Unloaded squats, with a 2s pause at the bottom, were performed to examine a slightly different motion profile. The results are shown in Figure 5. Because of the pause this data looks a whole lot more messy, and will be more difficult to post-process.

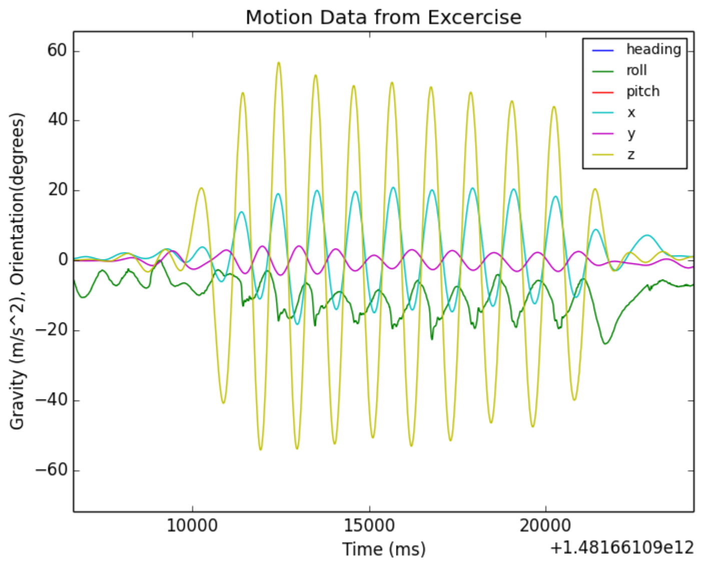


Figure 2. Motion profile of bench press warmup, one set of ten.

5. Next Steps

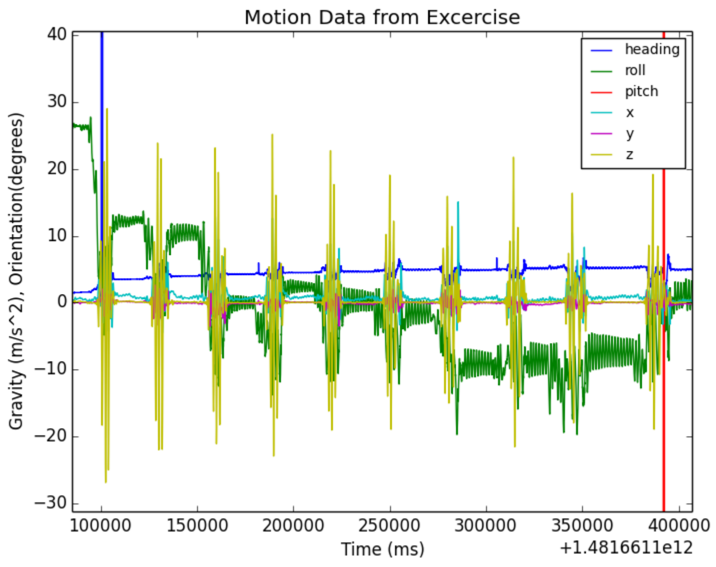
5.1. **Hardware.**

5.1.1. *Proto Validation.* Several more validation steps need to be performed to allow for a successful proto-2 release. These are shown in Table 2.

Table 2. Hardware validation todo list.

Task	Status
I2C Validation	Deprioritized. Hardware allows.
CP2102	Experiencing unexpected behavior
Si7021	Deprioritized. Harware allows.
Power Corners	Pending
USB-Only Behavior	Pending

5.1.2. *Proto 2.* The WiPy appears to have been a poor decision for this product. In addition to being somewhat fragile the documentation and userbase for the device is somewhat less than was expected. Furthermore, a WiPy 2 has been released, so support for the original will continue to get worse. A new microcontroller will be selected for version 2, ideally one allowing for Bluetooth Low Energy connections to a cell phone to allow use of the device beyond concept validation.



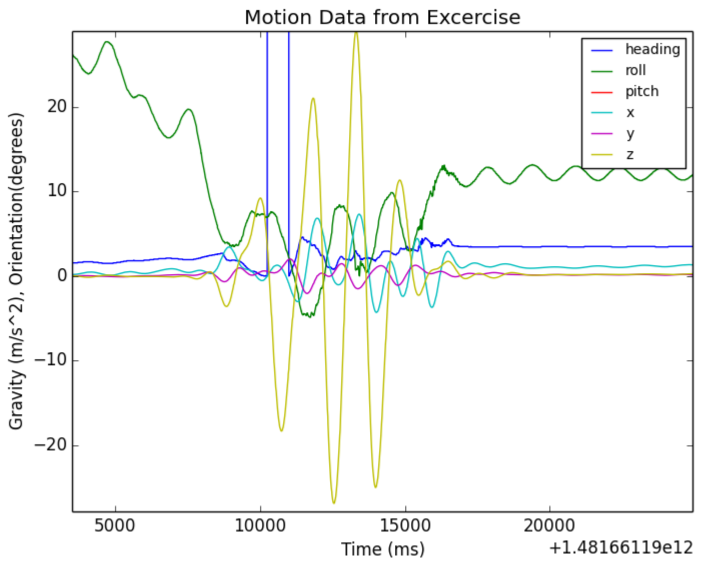


FIGURE 3. Motion profile of ten sets of three bench presses.

5.2. **Software.** Moving forward the data collection routine will be updated to support whatever platform is chosen for the next hardware revision. The motion processing module will be rewritten to use a Kalman filter to estimate true accelerations and orientations from

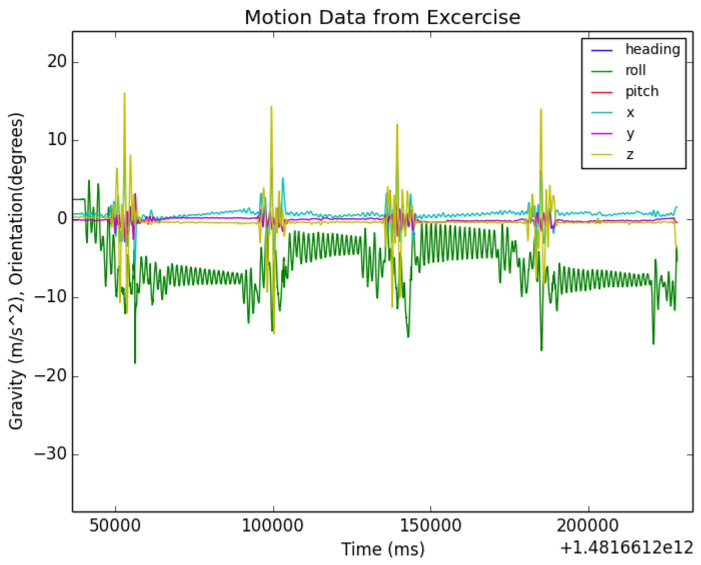


Figure 4. Motion profile of five sets of two bench presses.

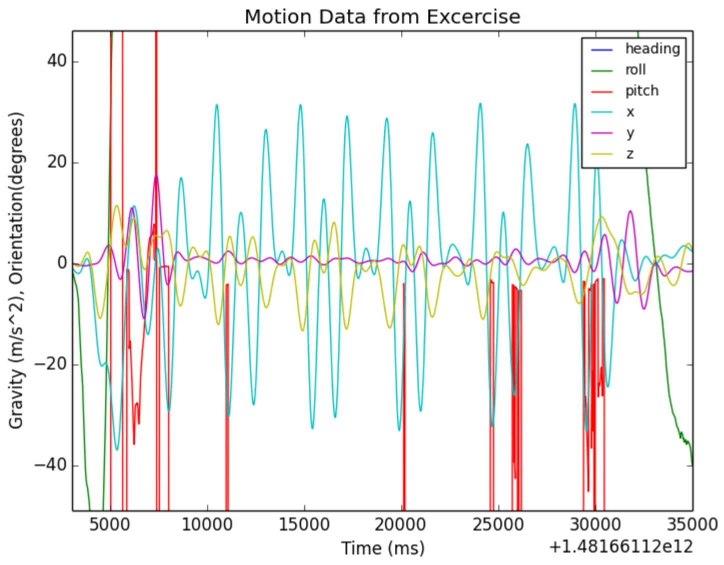
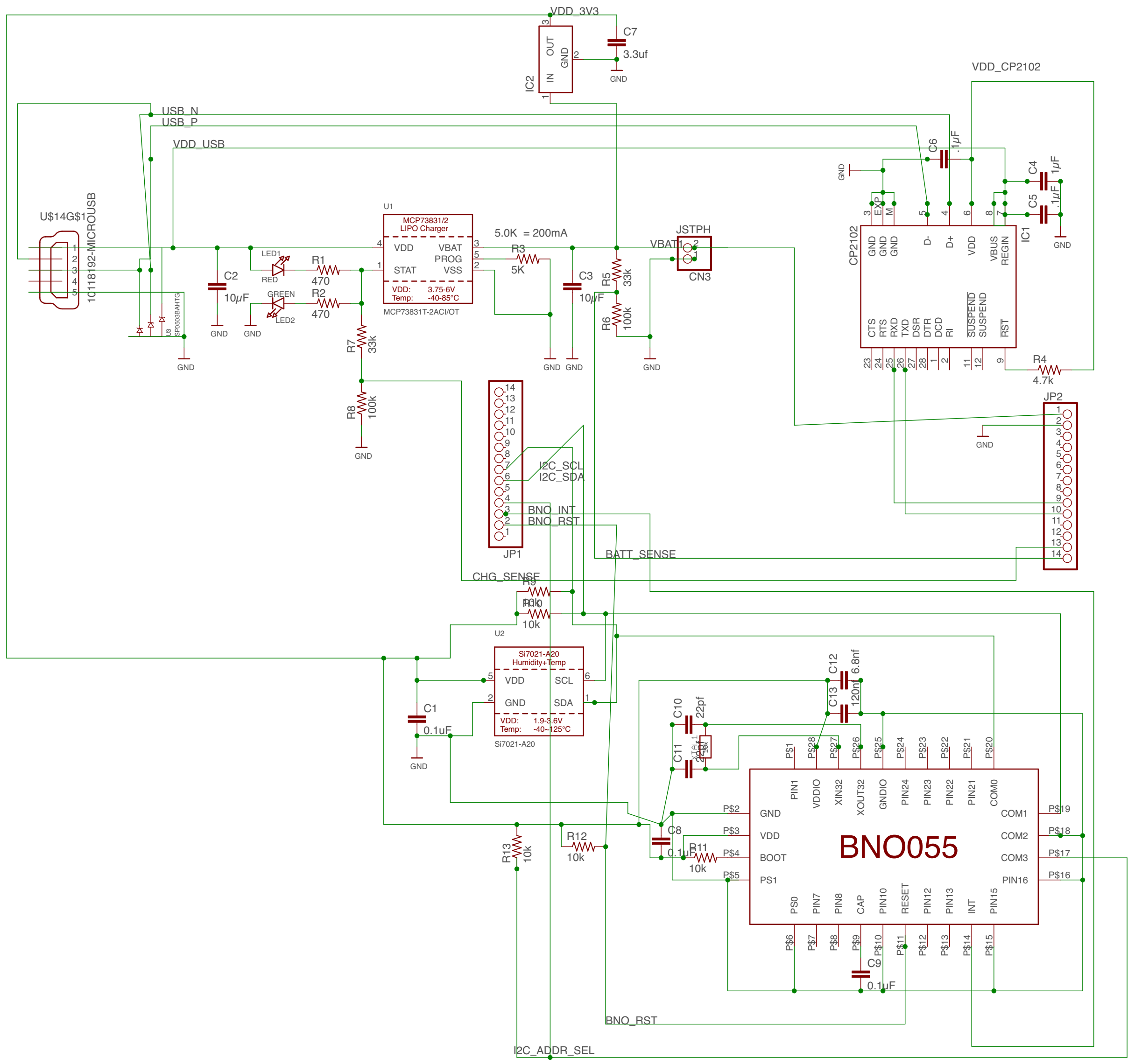


Figure 5. Motion profile of paused squats.

the noisy sensor output. This data will then be integrated into velocity and position, and finally processed to extract information of interest.

Appendix A. Schematic



12/13/16 11:16 PM f=0.80 /Users/cliff/Development/BarBuddy/board_cad/barbuddy.sch (Sheet: 1/1)

Appendix B. Code

Code can be found at `https://github.com/cliffbraun/barbuddy`