Equipack

Brenton Chasse CSE

Zach Boynton EE

Alex Nichols CSE

Colin Morrisseau EE

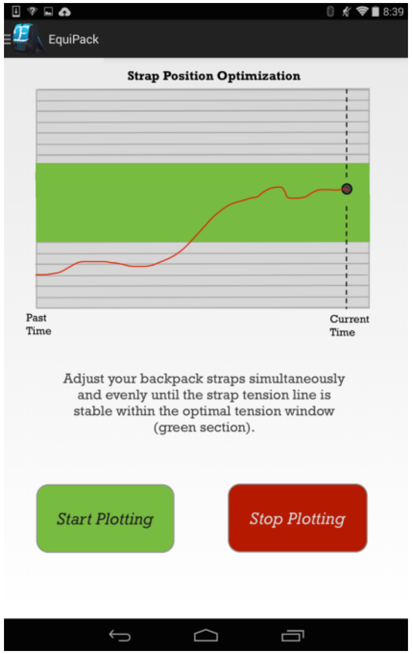
Advisor: Prof. Salthouse

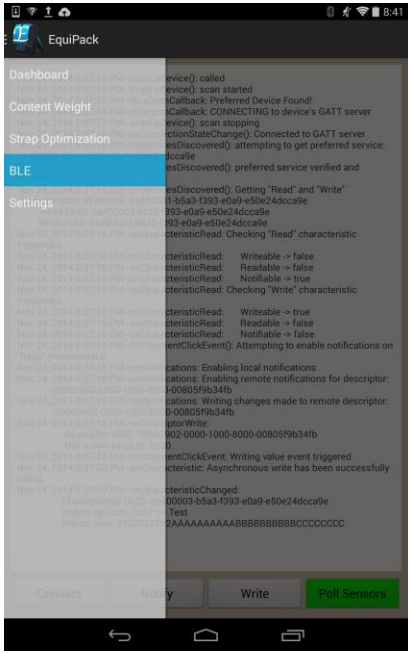
Health Risks from Backpack Misuse

- Misuse of backpacks
  - Improper pack positioning
  - Overloading pack
- +7,000 E.R visits annually
- ⅓ of 6th graders carry +30% of body weight
- Health Risks include:
  - Vertebral subluxation including herniation
  - Shoulder/neck stress

Brief Overview of Our solution

EquiPack provides a sensor network integrated into a backpack. Embedded hardware relays the sensor data to a mobile app, which provides a UI for displaying feedback on how to adjust the backpack to minimize health risks.

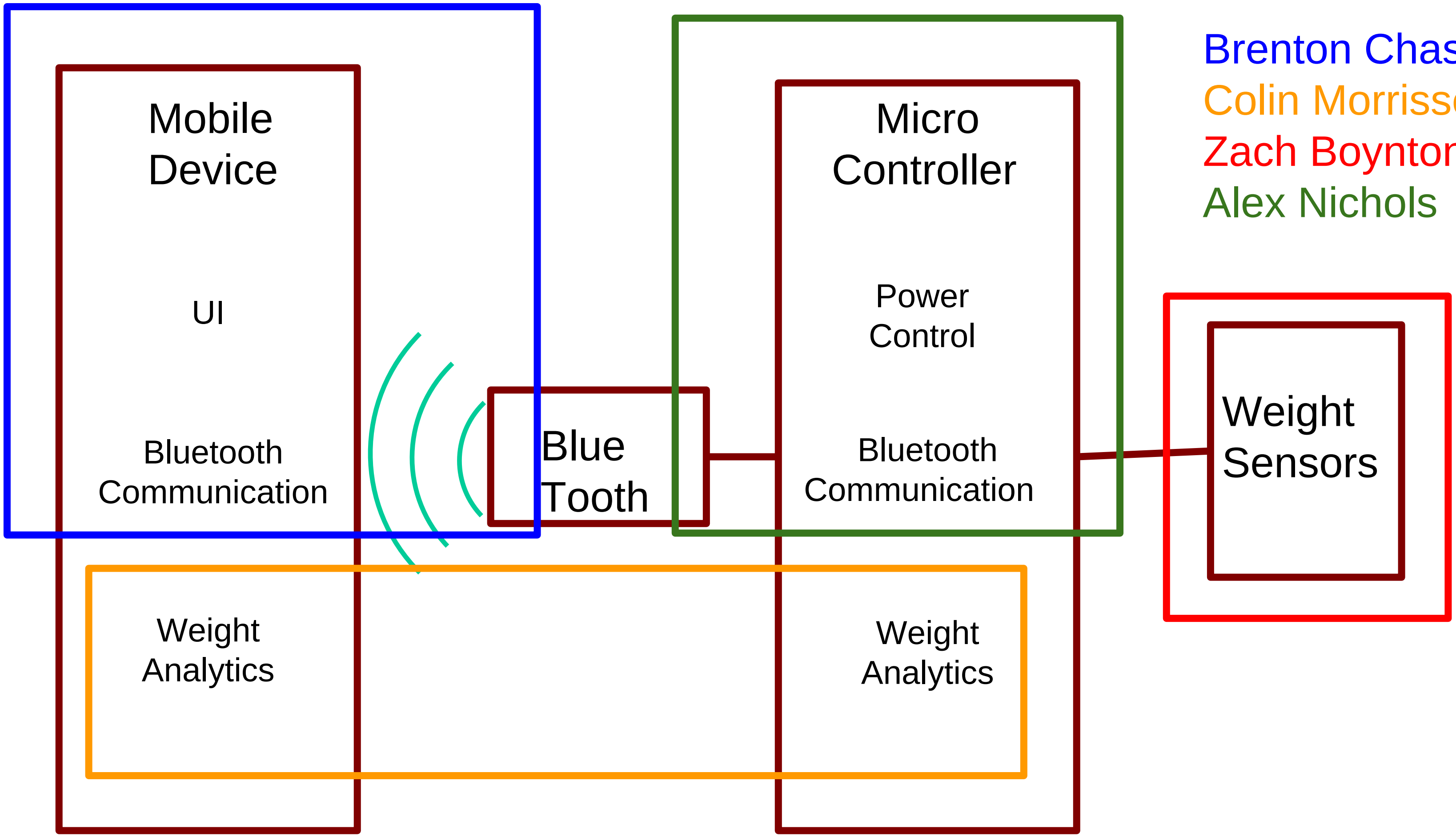




Source: dreamtime.com stockImages/VectorDrawings

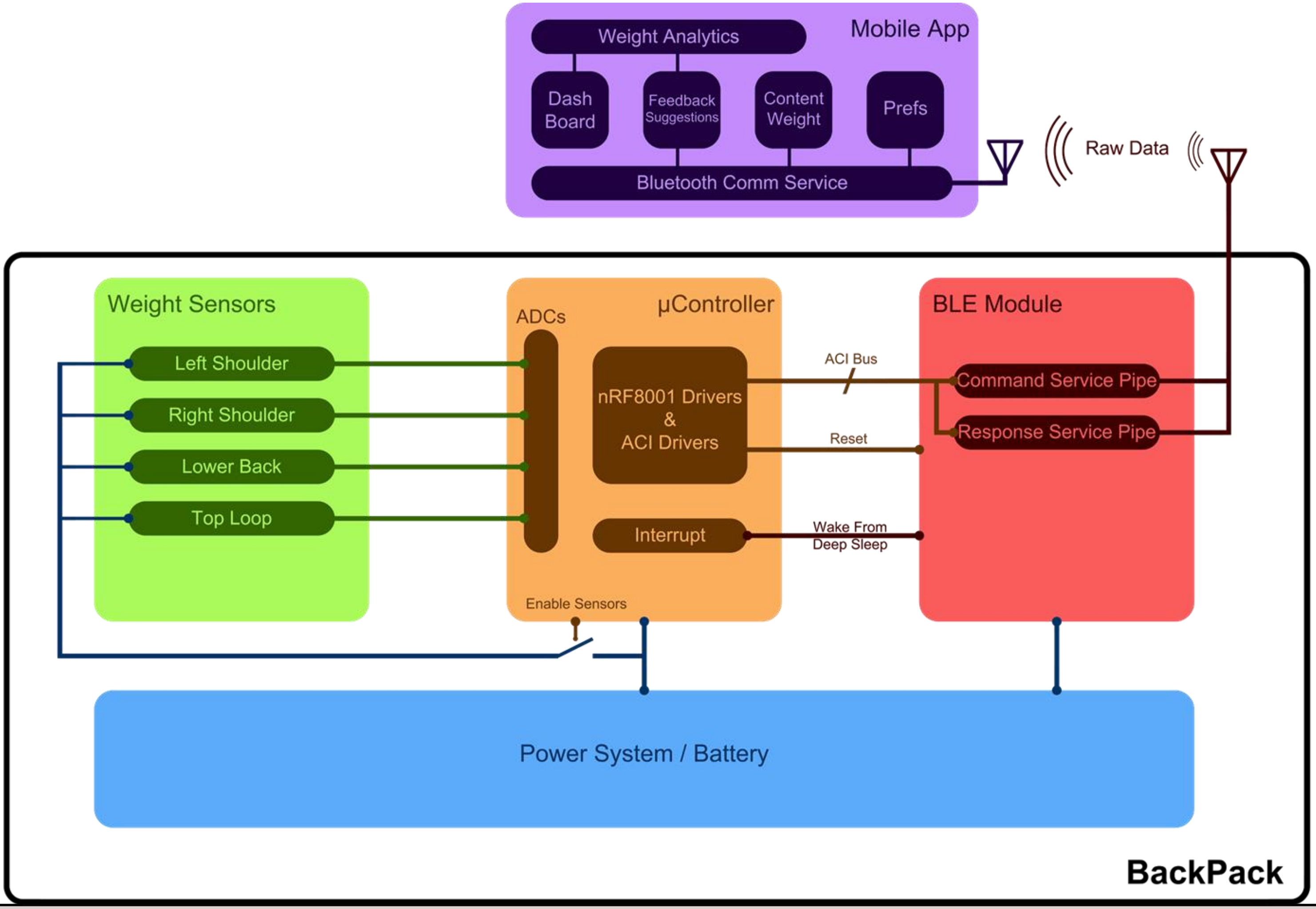
Advisor: Prof. Salthouse

Previous Block Diagram



Brenton Chasse
Colin Morrisseau
Zach Boynton
Alex Nichols

Redesigned Block Diagram



Proposed MDR Deliverables

- Weight sensor network converting physical force to a measurable signal
- Functional software Weight Distribution Model
- App UI interface w/ BLE sending and retrieving "data"
- First pass PCB design
- µController interfaced with:
  - Bluetooth transceiver module
  - Power systems

Proposed MDR Deliverables

- Weight sensor network converting physical force to a measurable signal
- Functional software Weight Distribution Model
- App UI interface w/ BLE sending and retrieving "data"
- First pass PCB design
- µController interfaced with:
  - Bluetooth transceiver module
  - Power systems

Proposed MDR Deliverables

- Weight sensor network converting physical force to a measurable signal
- Functional software Weight Distribution Model
- App UI interface w/ BLE sending and retrieving “data”
- First pass PCB design
- µController interfaced with:
  - Bluetooth transceiver module
  - Power systems

Proposed MDR Deliverables

- Weight sensor network converting physical force to a measurable signal
- Functional software Weight Distribution Model
- App UI interface w/ BLE sending and retrieving “data”
- First pass PCB design
- µController interfaced with:
  - Bluetooth transceiver module
  - Power systems

Proposed MDR Deliverables

- Weight sensor network converting physical force to a measurable signal
- Functional software Weight Distribution Model
- App UI interface w/ BLE sending and retrieving “data”
- First pass PCB design
- µController interfaced with:
  - Bluetooth transceiver module
  - Power systems

Proposed MDR Deliverables

- Weight sensor network converting physical force to a measurable signal
- Functional software Weight Distribution Model
- App UI interface w/ BLE sending and retrieving "data"
- First pass PCB design
- µController interfaced with:
  - Bluetooth transceiver module
  - Power systems

Proposed MDR Deliverables

- Weight sensor network converting physical force to a measurable signal
- Functional software Weight Distribution Model
- App UI interface w/ BLE sending and retrieving "data"
- First pass PCB design
- µController interfaced with:
  - Bluetooth transceiver module
  - Power systems

Proposed CDR Deliverables

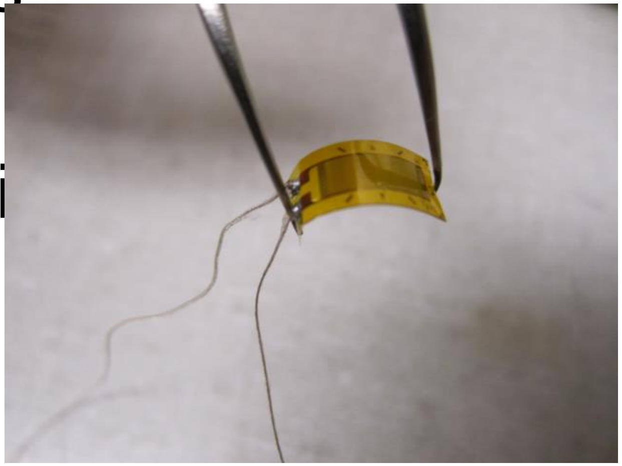
- Demonstration of Complete System Functionality
  - show integration between all subsystems
  - Show implementation of a battery powered system
  - Mobile application has UI elements to display feedback
    - BLE
    - Text Alerts
  - Show backpack can provide all core functions

Weight Sensors

- Prior Requirements:
  - 0-100lb weight range
  - 1lb granularity
  - environmentally insensitive
- Updated Requirements:
  - Same as previously stated with the additions of: Robust wiring, insensitivity to wiring contacts

Weight Sensors: Completed Tasks

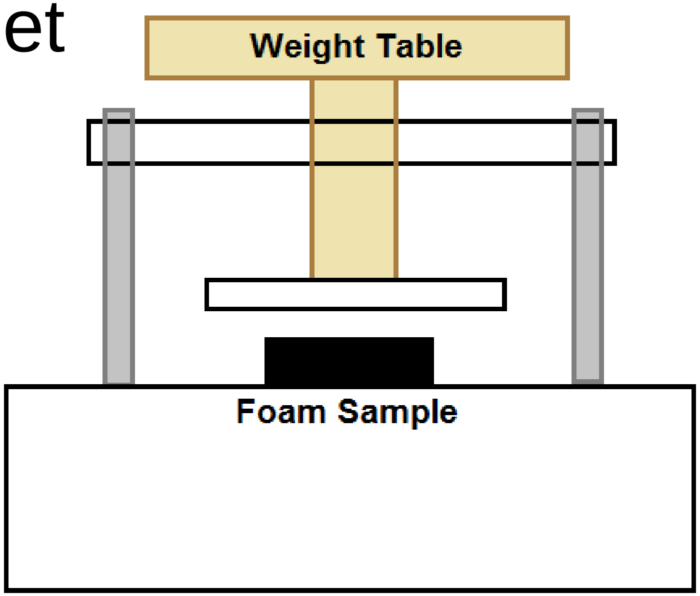
- I did tests to examine range, sensitivity, and repeatability of various sensor configurations
- Strain gauge and capacitive sensors had little change in physical properties
- Piezo sensors would not work easily for stati measurements



Source: http://www.ndsu.edu/pubweb/~braaten/research.html

Weight Sensors

- Conductive foam was picked for final sensor
  - Cheap
  - Easily Manufactured
- Current work has been to get clean readings from foam
  - Foam is sensitive to the contacts made
  - Foam can be modeled as an RC network

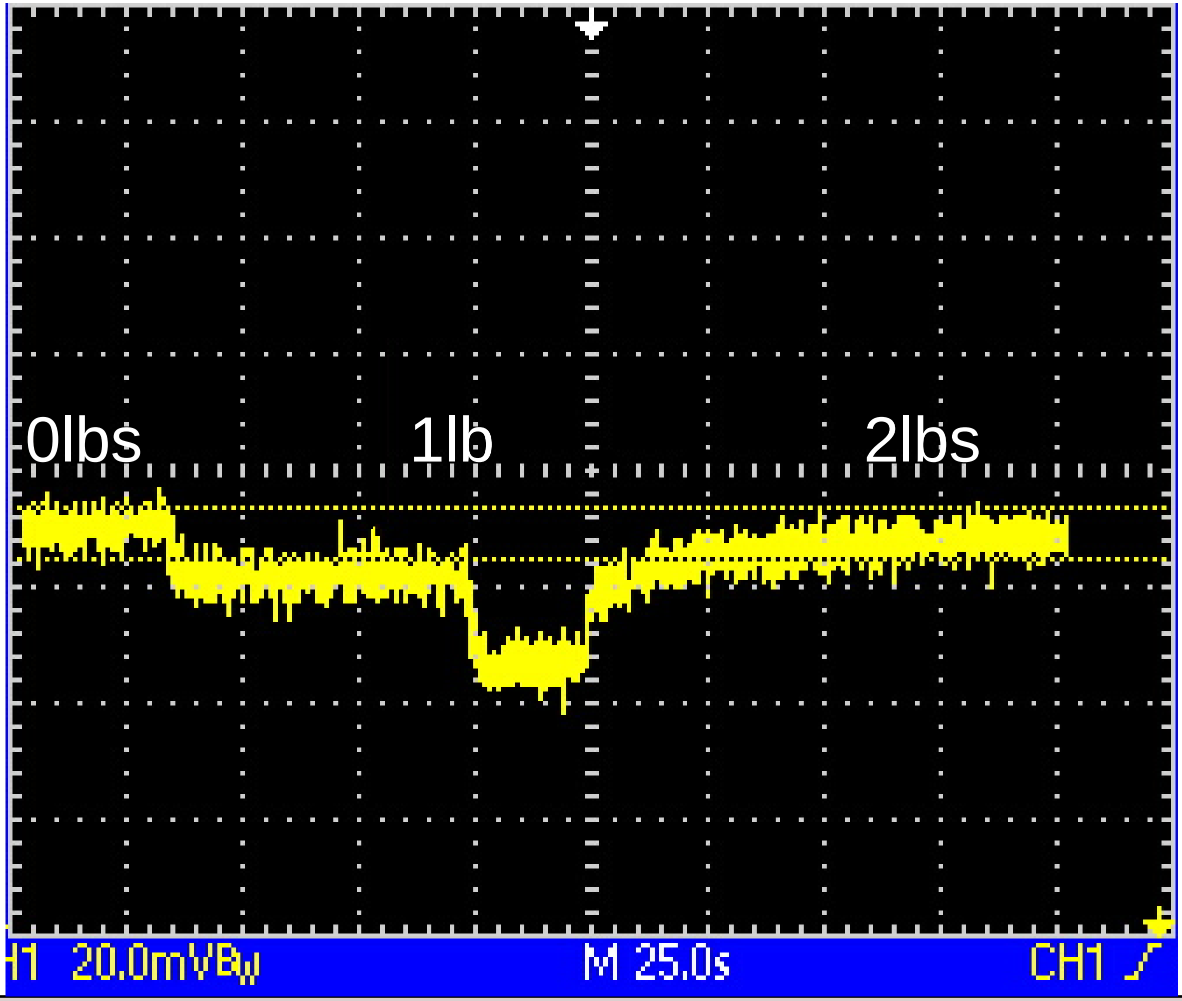


Weight Sensors

Example:

Values are noticeably different (~10mV/lb) for 1 lb increments. Measurements also return back to their initial conditions.

This will be demonstrated live at the end of the presentation

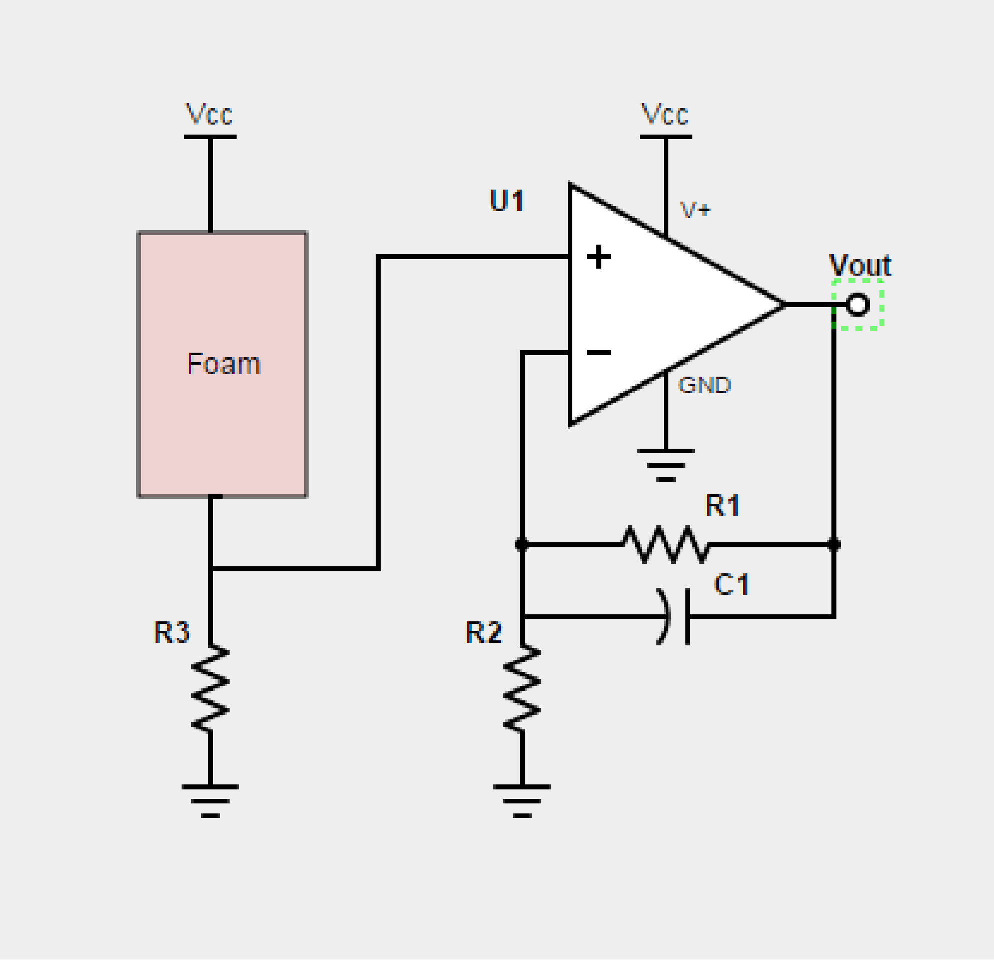


Weight Sensors: Schematic Diagram

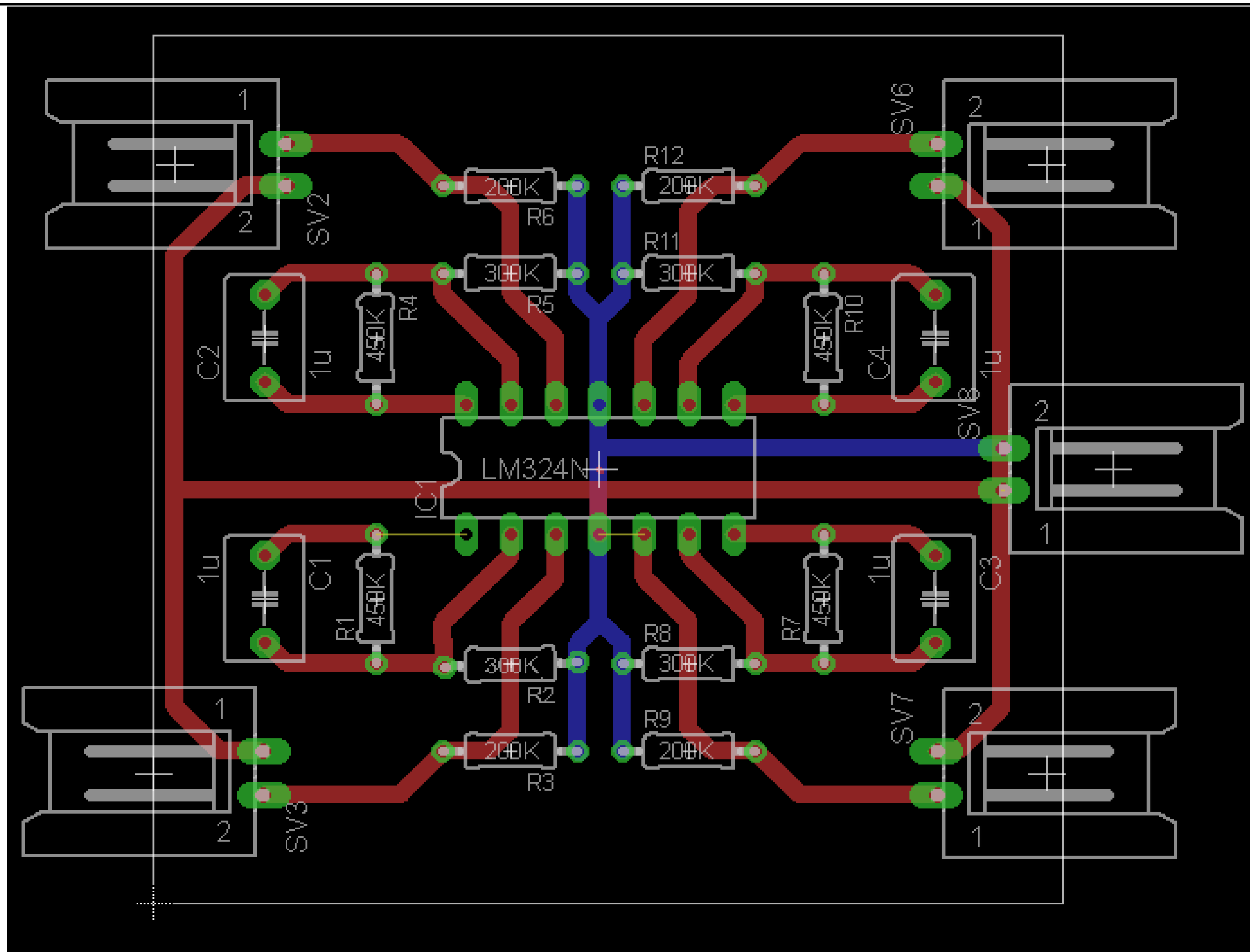
R1=450K
R2=300K
R3=680k
C1=33nF
Opamp LM324N
Vcc=5V

$$H(S)=\frac{R_1+R_2+R_1R_2CS}{R_1R_2CS+R_2}$$

P_Diss = 687.50 mW per sensor

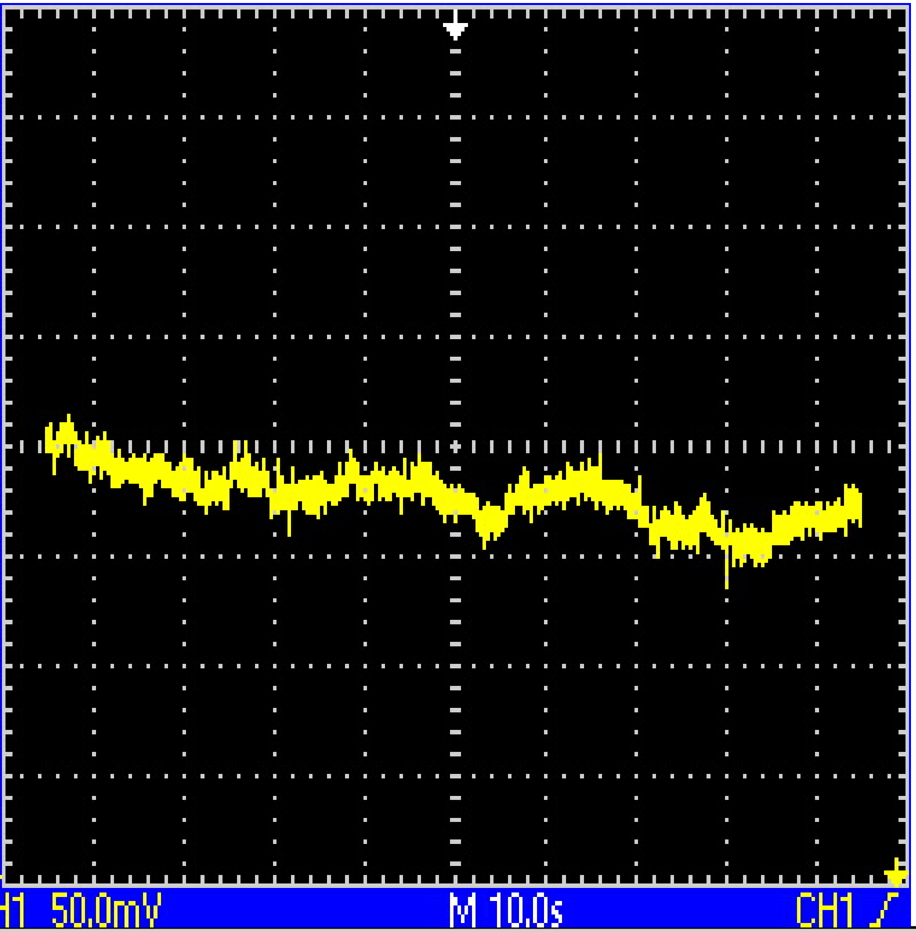


Weight Sensors: PCB Layout

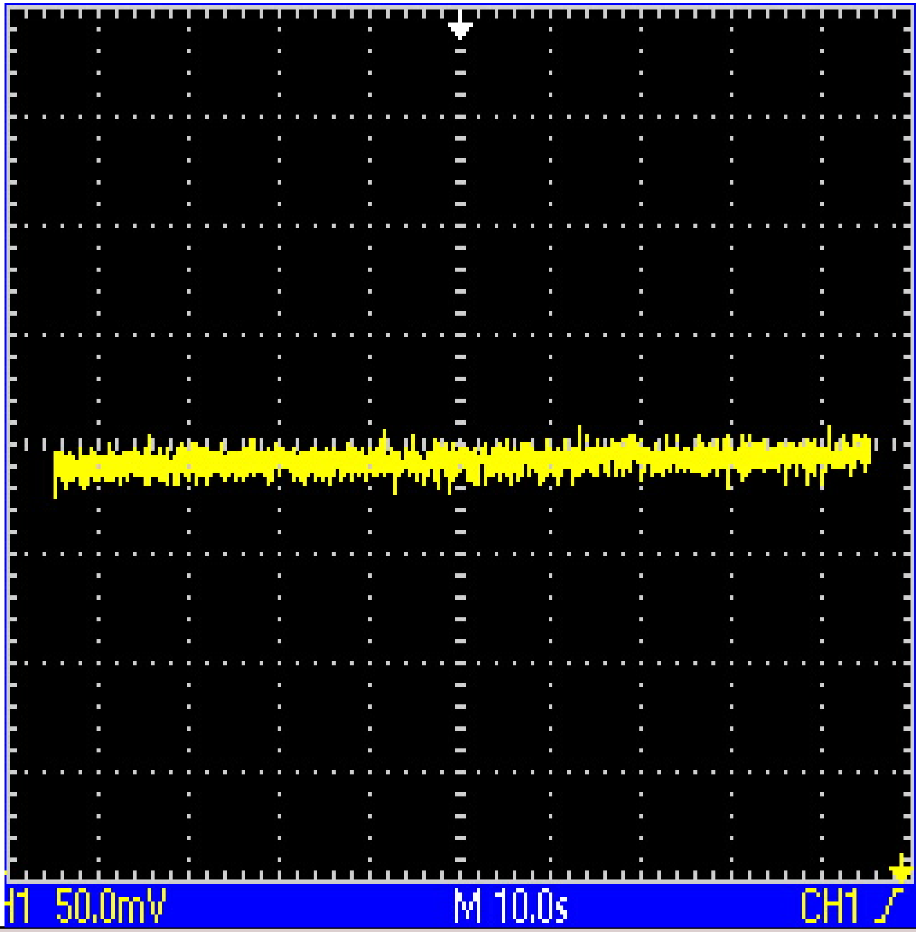


Weight Sensors: Contact

Contacts made with inserted wire:



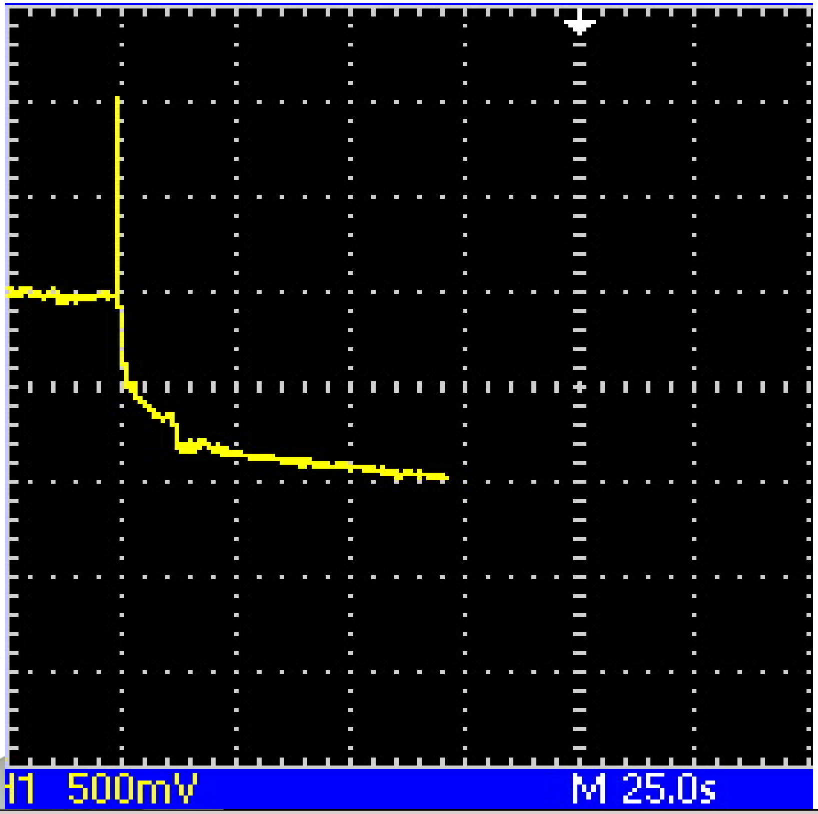
Contacts made with screws and washers:



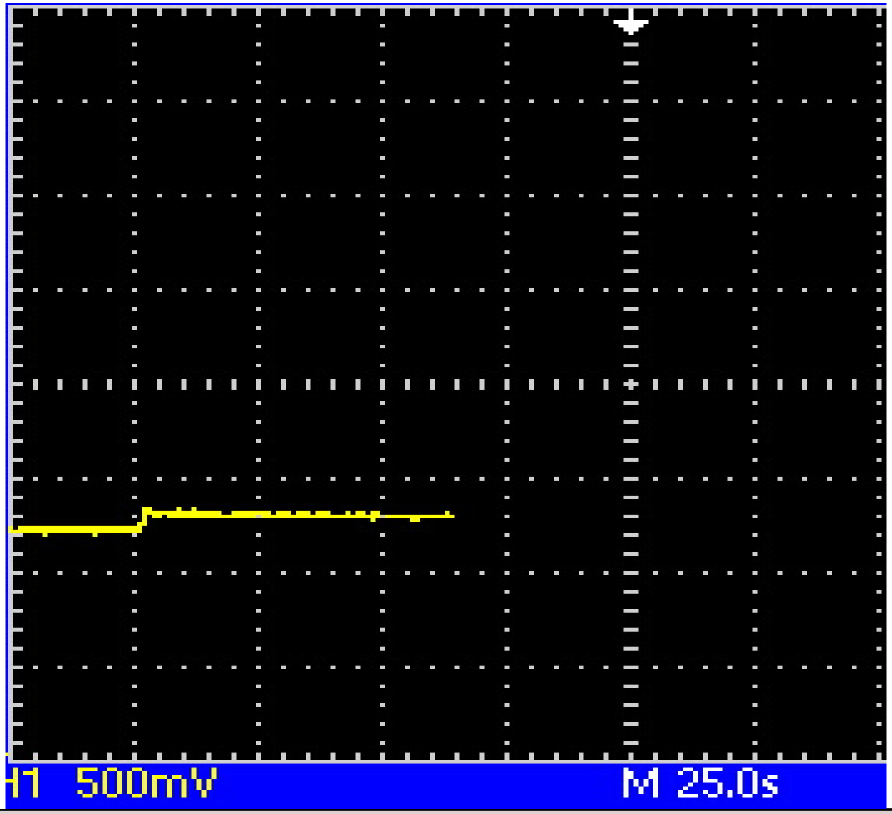
Weight Sensors: Contact

The measurement to the left is nonsensical as resistance, and consequently voltage should Increase!

Contacts made with inserted wire:

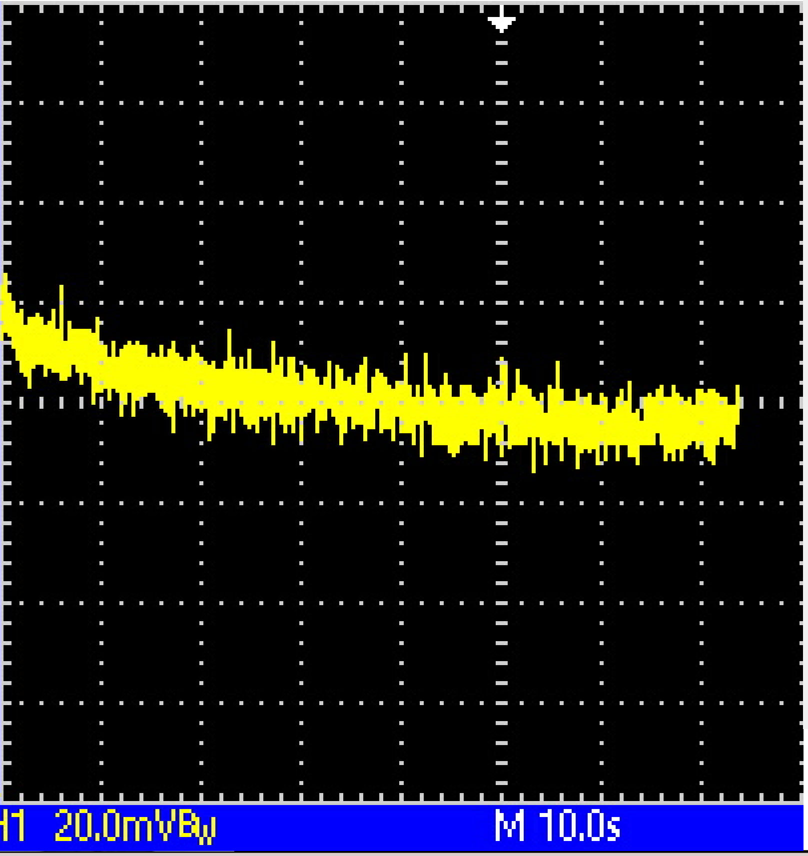


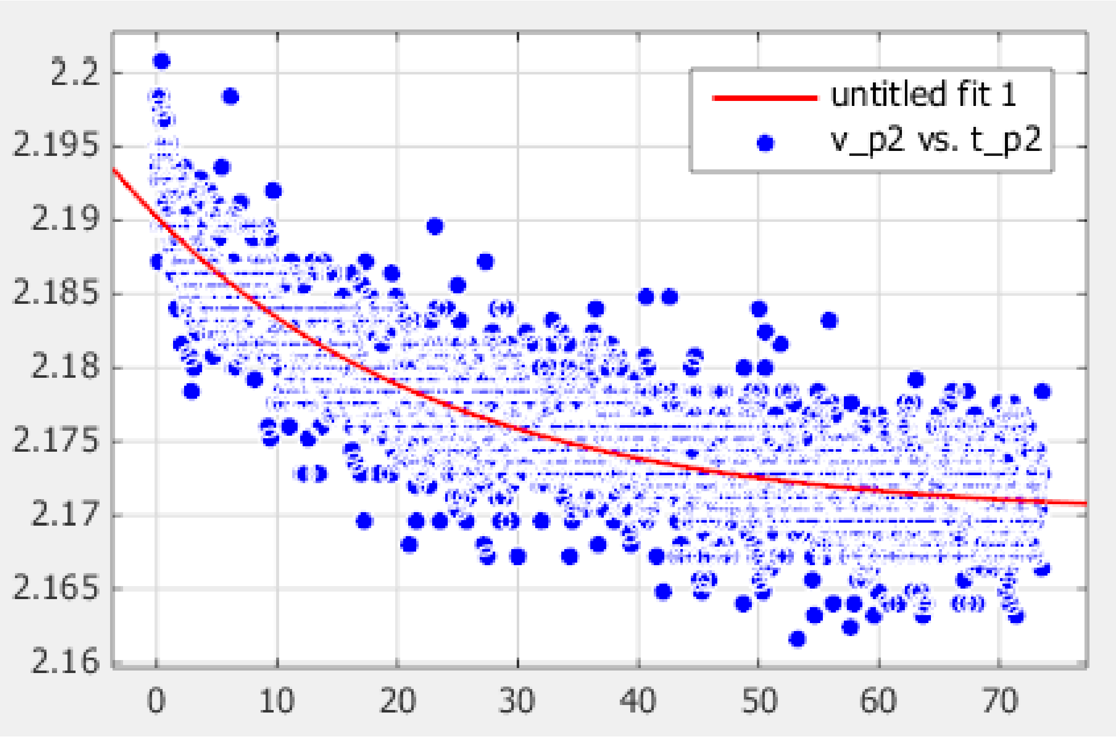
Contacts made with screws and washers:



Weight Sensors: Foam Modeling

The foam acts as an RC network and so requires time to settle into a steady state value.



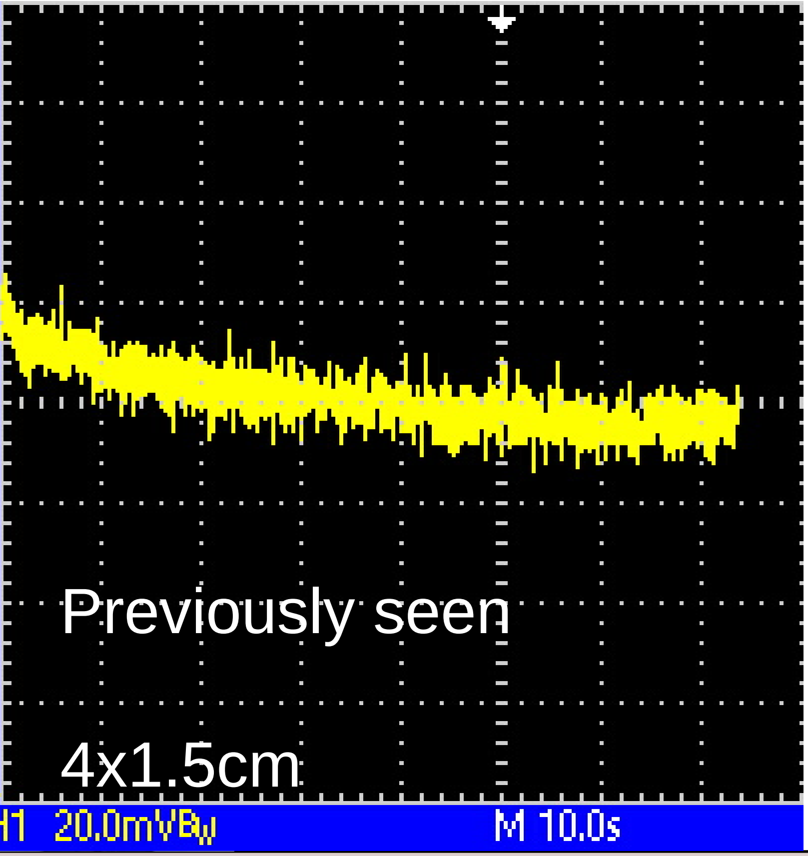


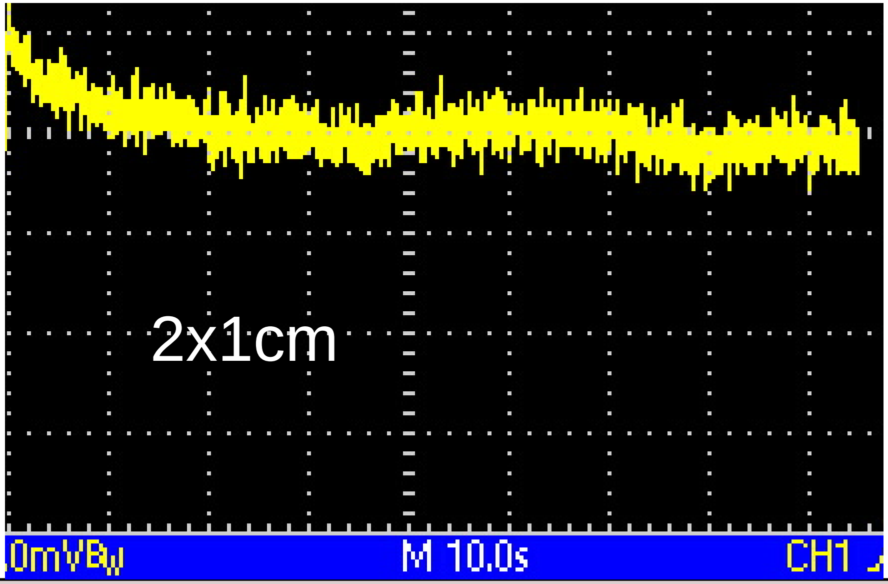
Fit line is of the form A*(exp(-B*t))+C in this case A is the initial value, b is 1/RC and C is the steady state value

Weight Sensors: Values for RC

The foam behaves as an RC network. The values of R and C change with physical dimensions

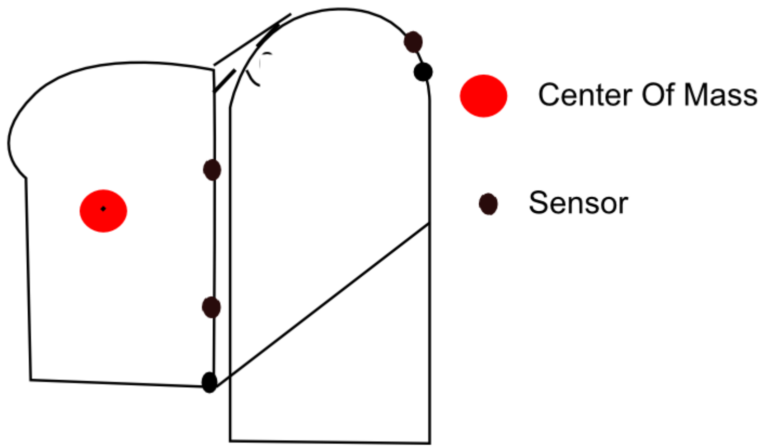
Dimensions	1/RC
4x1.5cm	0.4106
2x1cm	0.1089





Weight Analytics

- Subsystem Goals:
  - Determine Center of Mass
  - Determine Total Weight
  - Verify sensor locations
  - Determine algorithm for strap adjustments



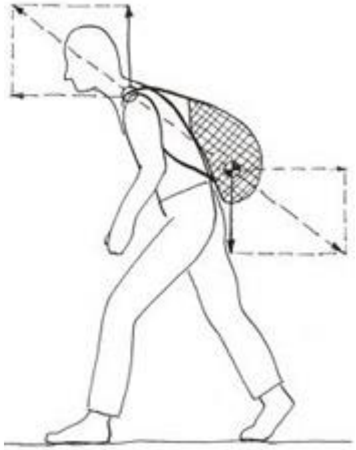
- Challenges
  - Verifying algorithms without being able to physically modify the system

Colin Morrisseau

Advisor: Prof. Salthouse

Weight Analytics

- Uses for analytics
  - Center of Mass determines forward lean. can be set to a threshold to prevent spine problems. utilizes the back and lower strap sensors
  - users will be recommended to only carry a percentage of their body weight from the total weight. utilizes the shoulder straps.
  - Strap adjustments aim to decrease the use pressure sensors on shoulders

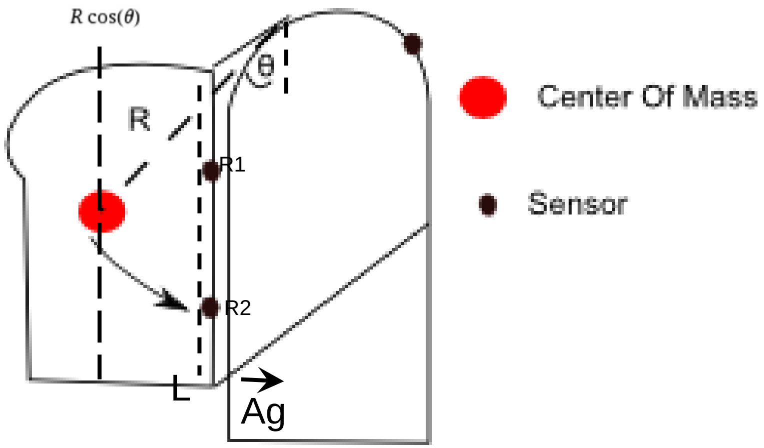


Colin Morrisseau

Advisor: Prof. Salthouse

Weight Analytics: Center of Mass

- By using pressure sensors along the back, we can determine the reactionary force that the backpack exerts at its surfaces.
- These forces on the back come from the backpacks pull from gravity and the fixed point of the pack at the top of the shoulder.
- If we assume all mass is located at the center of mass (an untrue but necessary assumption), we can determine the y-plane the center of mass is located on.
- Verified equation using bullet physics engine in Blender



Final Equation:

$$R\cos(\theta) = \frac{L^3}{2\,m\,g}\left(\frac{p_1}{r_1^2} + \frac{p_2}{r_2^2}\right) + A$$

A is a constant determined by strap tension

torque = mRgsin(theta)

Colin Morrisseau

Advisor: Prof. Salthouse

Weight Analytics: Optimization

- The Optimization algorithm is based off of a minimization function for the strap pressure on the shoulders
- Strap location can be determined by the maximum force on the upper or lower shoulder sensors. (exact ratio requires physical testing)
- Left/Right symmetry is chosen by deciding whether the left and right sides are balanced and adjusts straps accordingly.

Algorithm

```
while( abs(left - right) > minimum balance threshold )
           loosen higher pressure strap until equal;
//determine strap location by checking pressure on shoulders
if lower strap sensors < upper strap sensors
           set strap location to high
else set strap location to low

while(max(shoulder pressure at t+1) < max(pressure on shoulder)) at t)
           if strap location == low
                      tighten both straps
        else loosen both straps
           if any strap is above a safety threshold
              loosen both straps
              break;
```

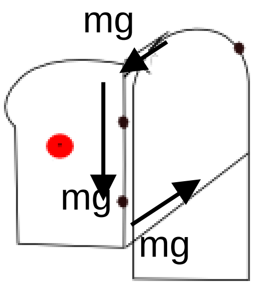
Colin Morrisseau

Advisor: Prof. Salthouse

Weight Analytics: Total Weight

- To determine total weight an additional sensor connected to the strap is required
- System can be thought of as a simple pulley because the mass is all in one location and friction is negligible
- weight is two times the strap measurement
- Verified by taking apart a luggage scale and inserting it in between the straps

Test load cell taken from luggage scale

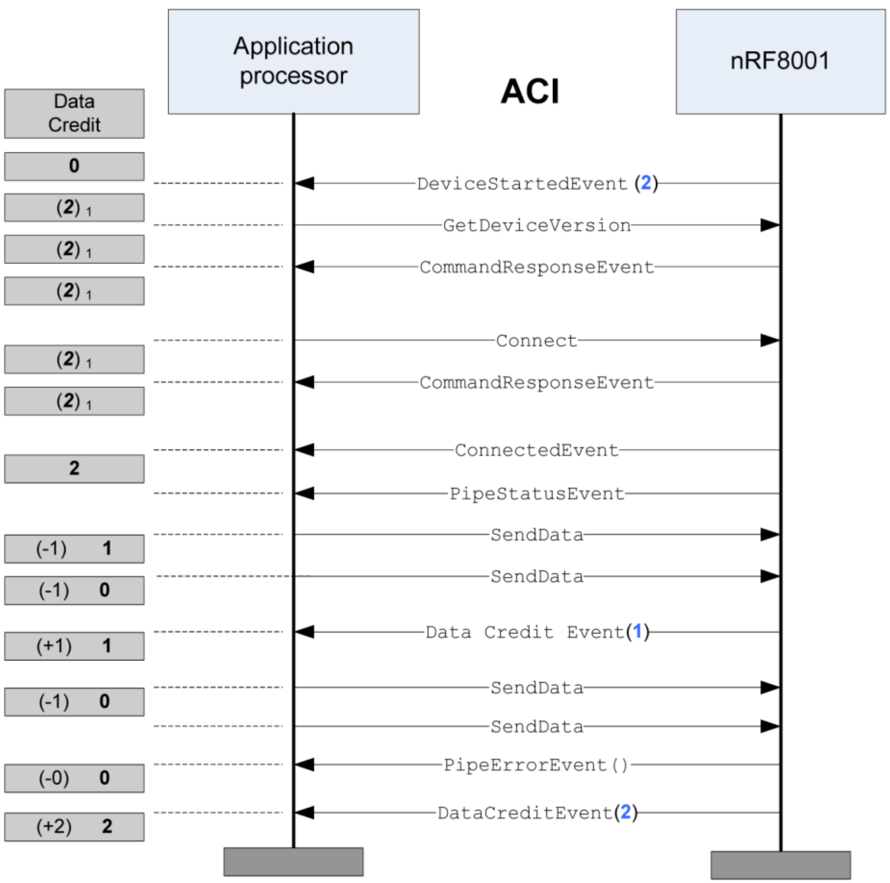


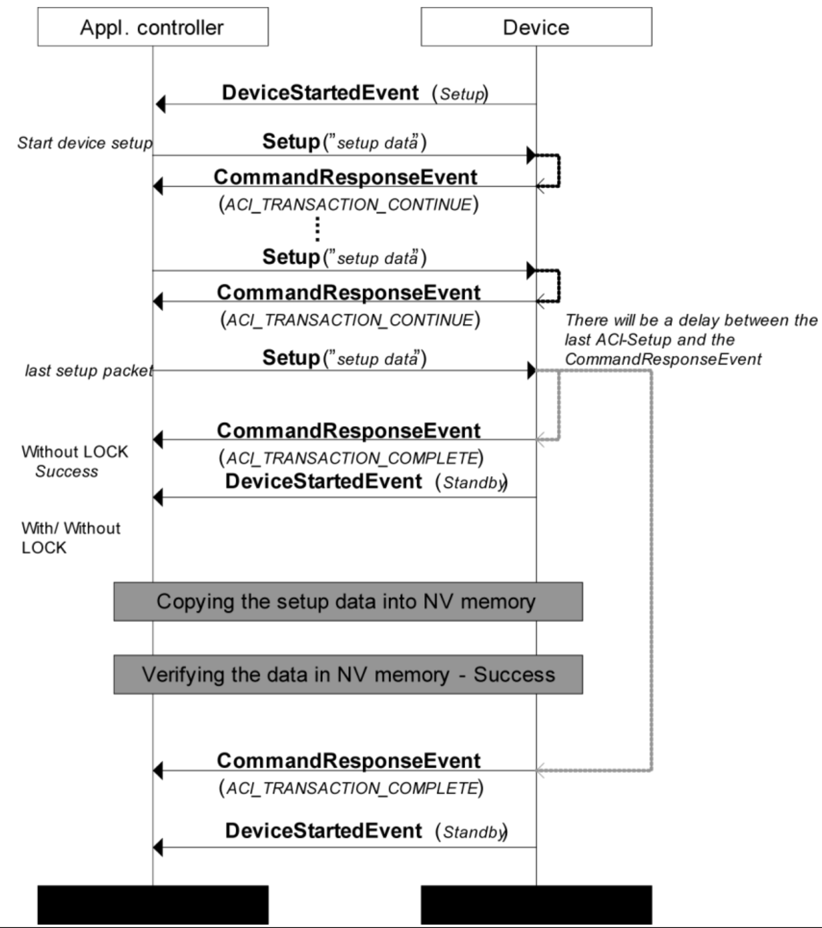
µController and Broadcast

- Previous requirements:
  - Low Power (10mA draw)
  - More than 8 ADCs
  - Bluetooth Module Implements Full BLE Gatt Server
- Hardware Choices Review
  - LPC824M from NXP Semiconductors (µController)
  - NRF8001 from Nordic Semiconductor (BLE Module)

µController and Broadcast

- Bluetooth-µController Communication





µController and Broadcast

- Challenges Faced
  - Challenge: Low Priority ADC interrupt not firing during BLE communication
    - Solution: *Interrupt Active Assert Register* (IAAR) is checked until interrupt state becomes active
  - Challenge: Digital Outputs could not drive pins on external Bluetooth Module
    - Solution: use digital output to drive non-inverting buffer, which in turn drives Module’s pins

µController and Broadcast

- Demonstration
    - To Demonstrate the functionality of the µController and the Bluetooth Module, we will show the BLE peripheral pair and connect with the Android App, and perform two operations:
        - Echo User Input
        - Stream raw sensor data from the µController to the Phone

Alex Nichols

Advisor: Prof. Salthouse

Waterproofing Options

- NeverWet Hydrophobic Coating (Rustolium)
  - Light, less bulky/easier to repair than epoxy
  - Potentially better heat dissipation
- Epoxy
  - More waterproof
  - Hard to perform repairs once coated
- Silicone and other rubberized coatings
- Shrink Tubing: provide additional protection around wires and solder joints

Image Source: http://www.rustoleumspraypaint.com/neverwet-faqs/

Mobile Application

- Prior Requirements
  - Secure data storage & transfer
  - Intuitive UI
  - Bluetooth Low Energy
  - Send text alerts
- Additional Requirements:
  - Expandable code base
  - Persistent customizable preferences
  - Reliability (error catching)
- Requirements Achieved
  - Secure data storage
  - Core interface/navigation for intuitive UI
  - Implementation of BLE stack
  - Persistent preferences

Brenton Chasse

Advisor: Prof. Salthouse

Mobile Application: Challenges

- First major challenge:
- Using the Android Bluetooth API and protocol
- Un-thrown exceptions within the stack
- Solution: Error handling and better understanding of how the bluetooth stack works.
- Second Major Challenge:
- Implementing the UI in such a way that sections of the UI can be reused, and all parts of the UI can talk to one main
- Solution: Using a fragment-activity approach rather than a view-activity approach.

Mobile Application: Design choices

- Activity:
  - Can be thought of as a “main”
  - Provides a screen that the user can interact with
  - Using one activity since all content is tightly bound internally
- Fragments vs. Views:
  - Represents a portion of the UI and it’s behavior
  - Added or removed while activity is running
  - Better use of screen real estate on large devices

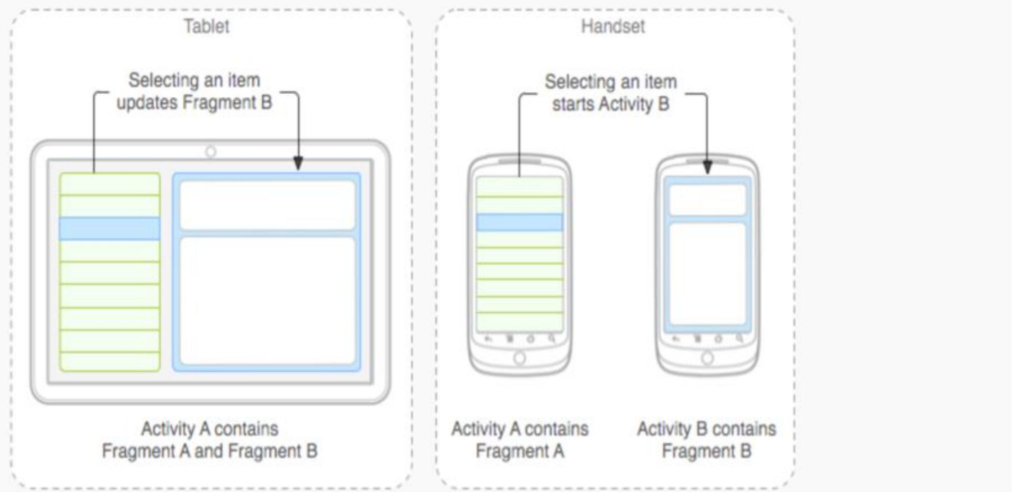


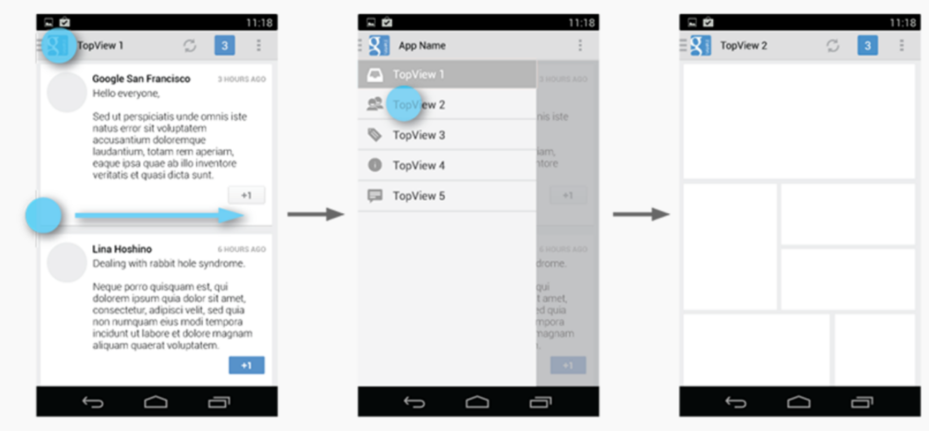
Figure 1. An example of how two UI modules defined by fragments can be combined into one activity for a tablet design, but separated for a handset design.

Image Source: developer.android.com

Mobile Application: Design choices

- Why pick Navigation Drawer as top-level navigation?
  - Suggested by Google if app has:
    - +3 top-level views (Can be used with Fragments)
    - Views are not directly related to one another (from the user's perspective)
- Preferences
  - Enable me as a developer to implement a security protocol
  - Can enter unique data about the user's pack
  - Preferences persist over multiple lifecycles of the app

Navigating with a Navigation Drawer

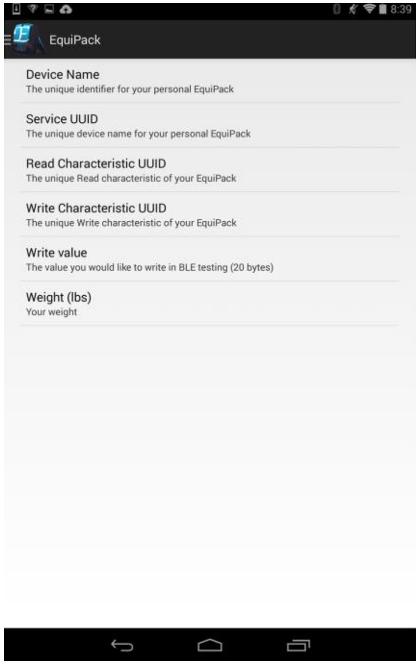


The user can open the drawer panel by touching the navigation drawer indicator.

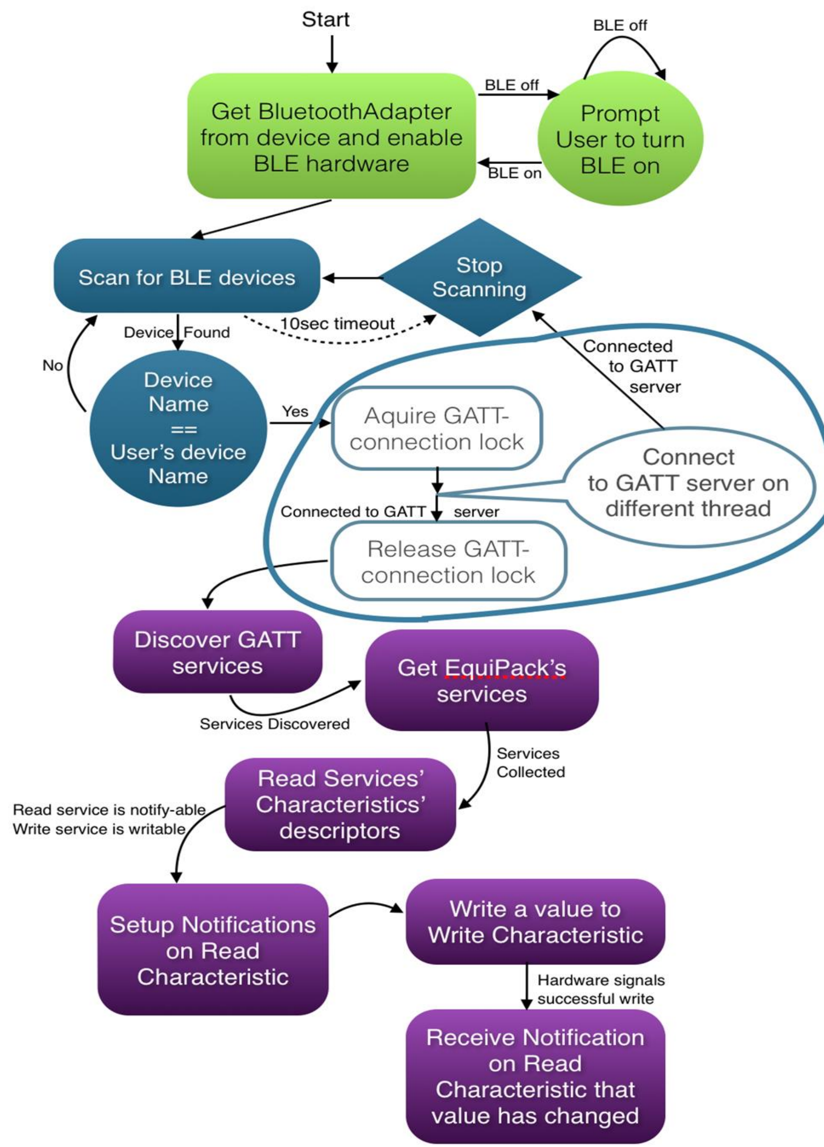
Source: android.developer.com

Mobile Application: Demonstration

- Top-level navigation can be performed through the Navigation Drawer.
- App preferences are persistent.
  (i.e.) If changed, it will be restored the next time the app is run
- BLE is Integrated with the embedded system:
  - Can poll the GATT server for ADC values
  - Can write to prefered characteristics
    - Expand to implement security handshake.



BLE state machine



Brenton Chasse

Advisor: Prof. Salthouse

Timeline/Schedule: Integration

	December	January	February	March	April
Zach	Sensor Housing	Finish Sensor Module	2nd Pass PCB		
			Power Systems		
Colin	Verify Models For Physical Sensors	Design Curve Fitting for input Data			
			Develop Java API for use in App		
Alex	PCB Design	Security		Control Battery & Power Systems	
		Selective Pairing Support	Power Saving Modes		
			2nd Pass PCB		
Brenton	UI Features				
		Send Alert Text			
			Develop API		
Integration	Weight Sensors with MCU		PCB Designs	App with W.A. API	
		Security Protocol between BLE module and App		All Subsystems into Back Pack	

Conclusion

- Questions?

Timeline/Schedule: Zach

-December: Sensor housing built to handle weight requirement. Start to integrate with microcontroller.

-January: Finish weight sensor module. Continue with microcontroller integration.

-February: Begin power systems work. Begin 2nd pass PCB if required.

-March: Begin integration power systems and sensors into bag.

-April: Final debugging and integration

Timeline/Schedule: Colin

-December: verify models with physical sensors
-January: design curve fitting algorithm to speed up the response time of the sensors
-February: continue previous as necessary
-March: develop API for digital implementation in the smartphone app
-April: final debugging and integration

Timeline/Schedule: Alexander

-December: Integrate µController PCB Design with Weight Sensor PCB design

-January: Keep track of Various Phones, integrate with NVM. Implement top-level encrypted communication with Android Phone

-February: Work On 2nd Pass PCB Design. Start Using Power-saving functionality on µController and BLE module to ensure optimal sleep schedule

-March: Begin integration into Bag; begin using battery for power

-April: Debugging and stability enhancements

Timeline/Schedule: Brenton

-December: More error handling, Start adding basic UI features

-January: Continue adding basic UI features, Sent text message to remote device upon a given condition. Implement top-level encrypted communication with µController

-February: Enhance appearance of UI features, Finish sending text message ensure solid stability of current features. Begin API as required features become defined.

-March: API for interfacing with the UI elements to display equipack calculations

-April: Defect/stability fixes, finish any tasks that have rolled over