Foot Stress and Blood Flow Sensing Under High-Loading during Hip Surgery

Sponsored by: Stryker Sports Medicine

PolyHIPS

Chris Cardenas: chriscardenas88@gmail.com
Esther Chen: echen4007@gmail.com
Claire Kingston: claire.kingston3@gmail.com
Katie Paswaters: kpaswaters95@gmail.com
Bri Rodebaugh: briannarodebaugh5@gmail.com

Statement of Disclaimer

Table of Contents

List of Tables

List of Figures

Executive Summary

The PolyHIPS team was tasked with integrating sensors into Stryker's hip arthroscopy boot that monitor blood flow and heel slip during hip surgery. During hip arthroscopy, the femur head is distracted from the acetabulum to create space in the joint for the surgeon to operate. This requires the use of up to 200 lbf. In order to maintain this space, the foot is held tightly in the boot. Complications can result from the foot being held too tightly in the boot during the procedure. The most common complication is neuropraxia, or temporary nerve damage. PolyHIPS proposes a solution that utilizes three types of sensors to monitor for nerve damage and heel slip. The final design includes three pressure sensors and a distance sensor that are all wired to a microcontroller (Figure 1). The microcontroller displays the sensor data on an LCD screen and transmits the data via Bluetooth to an app.

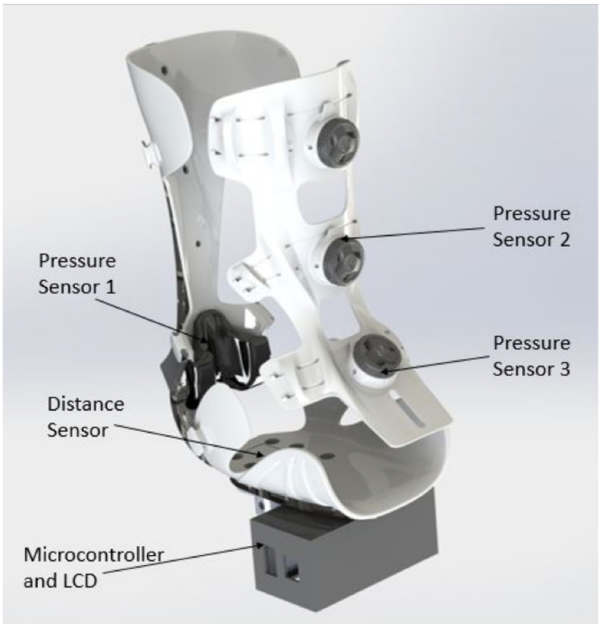


Figure 1. Overall Boot Configuration

Pressure sensors are used to avoid overtightening, which can mitigate the risk for nerve damage. The pressure sensors are Round Force Sensitive Resistors from Adafruit that will target key pressure points on the foot and ankle that are at risk during the procedure (Figure 1). Two sensors are attached to the BOAs located on the top of the foot and front of the ankle. The third pressure sensor is in the calcaneus pad on the back of the boot. The three sensors will beep when the correct tightness has been reached prior to distraction. If the calcaneus sensor loses pressure indicating the heel has slipped past the calcaneus pad, an alarm will sound to alert operating room staff. This will allow hospital staff to monitor the compressive force that the boot is inflicting on the nerves in the foot before and during the operation.

Heel Slip will be monitored by the Time of Flight sensor by Adafruit. This will be integrated into the heel of the boot and give a live readout of axial slipping of the foot (Figure 2).

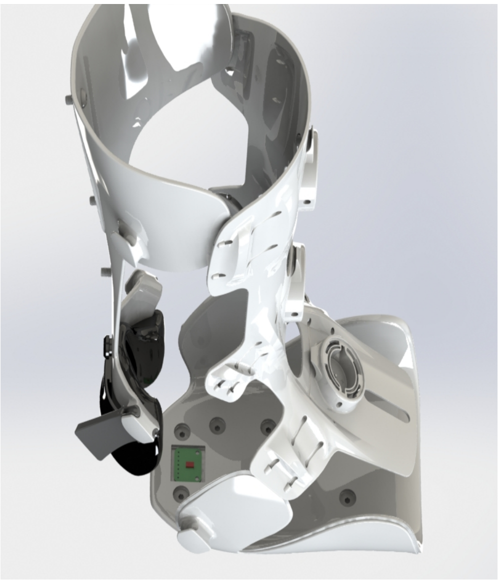


Figure 2. Distance Sensor Placement

All three pressure sensor and the distance sensor are attached to a Arduino Uno microcontroller. This is connected to a Bluetooth Low Energy module by Adafruit that wirelessly transmits data to a Windows tablet for hospital staff to monitor.

Chapter 1. Introduction

Although hip arthroscopy is one of the fastest growing surgical procedures in sports medicine, the method by which surgeons distract the hip causes complications during surgery and nerve damage after surgery. In order to create enough space between the femoral head and the acetabulum, the leg is held in place by a tightly fitted boot, and 200 pounds of force must be applied [1]. The two main complications that arise from this method are lack of blood flow to the foot and heel slipping in the boot [1]. Lack of blood flow during surgery can lead to neuropraxia, or temporary nerve damage, after surgery, and heel slipping can lead to loss of joint space during surgery [1]. These complications can be painful for the patient and can cause problems for the surgeon.

Stryker Sports Medicine has developed a boot that effectively distracts the hip during hip arthroscopy, but their goal is to improve the design by adding a method to alert surgeons of morbidity in the foot and heel slipping in order to mitigate surgical complications and postoperative pain for patients. The goals of this project are to design a method that collects and records data on blood flow and stress in a single output that is linked to a display in the operating room. Furthermore, the design should include an audio indicator that alerts the surgeon when pressure and heel slipping become critical. The design should be compatible with the current surgical workflow. The goal is to record and track the data at specified time intervals.

The overall objective of this project is to design a device that detects pressure and heel slip during hip arthroscopy so that the surgeon can correct the position and stress on the foot before it causes postoperative pain for the patient. Ideally, these additions should not only have an independent power source but also be able to transmit information wirelessly. Some customer requirements, specific to the physician implementing the surgery, include detection of heel slip, detection of pressure, ease of use, simple user interface, visual/auditory warning, spatial awareness, minimal change in procedure, affordable, sanitary, water resistant, wireless and sufficient lifespan. Additionally, the device should satisfy the patient's requirements of remaining safe and comfortable.

Customer requirements were evaluated for both the physician and patient. Engineering requirements for the project are listed in Table 1 and Appendix A outlines their relationship with the customer requirements.

Table 1. Engineering specifications for device

Spec. #	Parameter Description	Requirement or Target (units)	Tolerance	Risk	Compliance
1	Accuracy of Pressure Sensor	Yes	± 0.1 psi	High	I
2	Accuracy of Heel Displacement	Yes	± 1/4 inch	High	I
3	Boot Weight	5 lbs	Max	Low	I
4	Ability to be Cleaned and Disinfected	Yes	N/A	Low	D
5	Withstand Force	200 lbs	Max	High	T, A
6	Wireless range	15 ft	+/- 5 ft	Medium	D, A
7	Battery life	5 hrs	Min	Medium	T, A
8	Windows Compatible	Yes	N/A	Low	D, T

Risk is determined as low, medium, or high; high-risk being the most critical to quantify the success of the device. In order to judge if parameter description is met, listed under compliance method are the actions to be performed for that requirement: inspection (I), demonstration (D), test (T), and analysis (A). Inspection is the nondestructive examination of product using one or more of the five senses. Demonstration is the manipulation of the product to verify that the results are as planned. Test is the verification of the product using a controlled series of inputs to ensure the product will produce a specific outcome. Analysis is the verification of a product using models, calculations and testing equipment.

Heel displacement accuracy is aimed to be as precise as possible; however, the ¼" tolerance is representative of the width of the surgeon's finger. Surgeons currently use their pinky finger to measure the degree of heel slippage within the boot. The weight of the boot after the addition of sensors should not exceed 5 lbs, as per Stryker's current boot engineering requirements. Cleaning and disinfection of the device will be demonstrated by applying Sani-cloth Wipes post use; this ensures device is available for reuse. Presently, the boot is able to withstand a force of 200 lbs; the addition of sensors should not impede in that function. Wireless range must accommodate for various operating rooms. According to the

Facility Guidelines Institute, operating rooms are required to be a minimum of four hundred square feet. Rooms with more staff and equipment are required to be a minimum of six hundred square feet with a length of twenty-four feet [2]. To accommodate for large operating rooms, a wireless range of 15 ft would ensure the signal could be transmitted throughout the room. Battery life should well exceed that of surgery time. The typical length of hip arthroscopy surgery can last between 1-2 hours [3]. A 5-hour battery life should exceed the length of any surgery that extends past the normal time. Lastly, per Stryker's request and ease of use, these sensors must operate using Windows.

Chapter 2. Background

Hip Arthroscopy (HA) is a surgical procedure in which doctors are capable of viewing and treating hip issues all while keeping incision size limited. A camera is inserted into the hip joint, allowing doctors to assess the damage more easily and maintain a higher success rate than open surgery. The camera displays live images of the hip joint which the surgeon utilizes to guide the surgical instruments.

During Hip Arthroscopy, the hip joint is operated on. The hip joint is a ball and socket synovial joint formed between the acetabulum of the pelvis and the head of the femur (Figure 3).

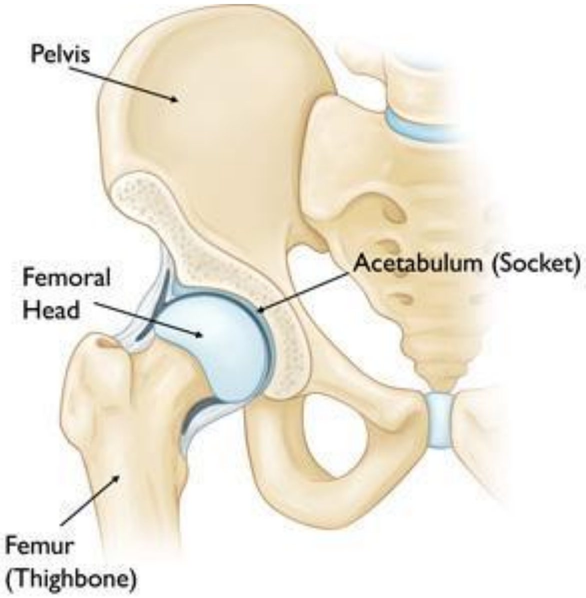


Figure 3. Anatomy of the human hip [4]

Hyaline cartilage lines both bone interfaces, allowing for the bones to glide past each other. Between the hyaline cartilage is synovial membrane which secretes synovial fluid to lubricate the joint. Ligaments and muscles hold the hip joint in place and prevent dislocation.

Before the surgeon can begin operating, the hip must be distracted, a process which separates the femoral head and acetabulum, creating space in the hip joint for the surgeon to work, as shown in Figure 4 [1].

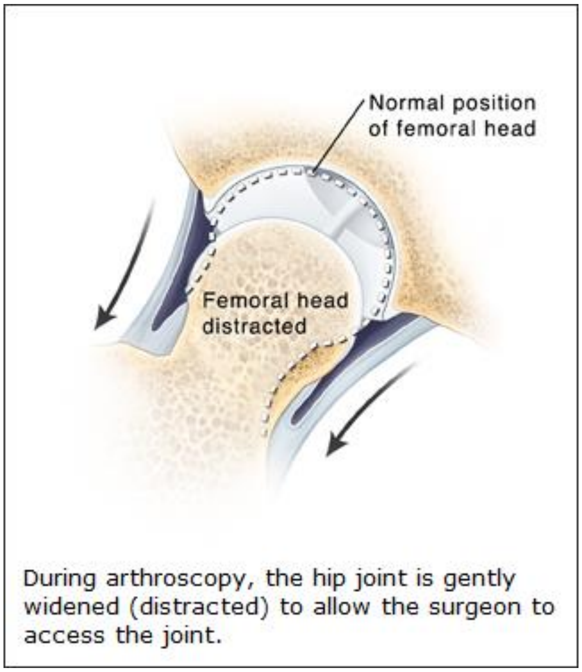


Figure 4**.** Normal vs distracted femoral head position [5]

In order to distract the hip, the doctor must apply at most 200 pounds of force to have enough room to progress [6]. The patient will be put under an anesthetic before the distraction process begins. Then the patient will have both feet secured for the duration of the operation. The non-operative leg is secured and positioned to be out of the way of the nurses and surgeon. The operative leg is secured more tightly to allow for the application of the distracting force. Once the feet are secure, the patient will be tilted backwards (head down, feet up) at a 10 to 15 degree angle [6]. This angle creates the most optimal positioning for hip distraction. The surgeon will use the distraction device, whether it be a ratcheting mechanism or a crank mechanism, to distract the hip. A C-arm x ray machine is placed over the patient in order to take images throughout the distraction process to ensure the joint is separating correctly (Figure 5). The doctor operates within a narrow gap between the femoral head and acetabulum, leaving minimal room for complications. Typically, this surgery takes around 2 hours [6]. However, this is not always the case considering the hip is often looked at as the most complex joint to arthroscope on.

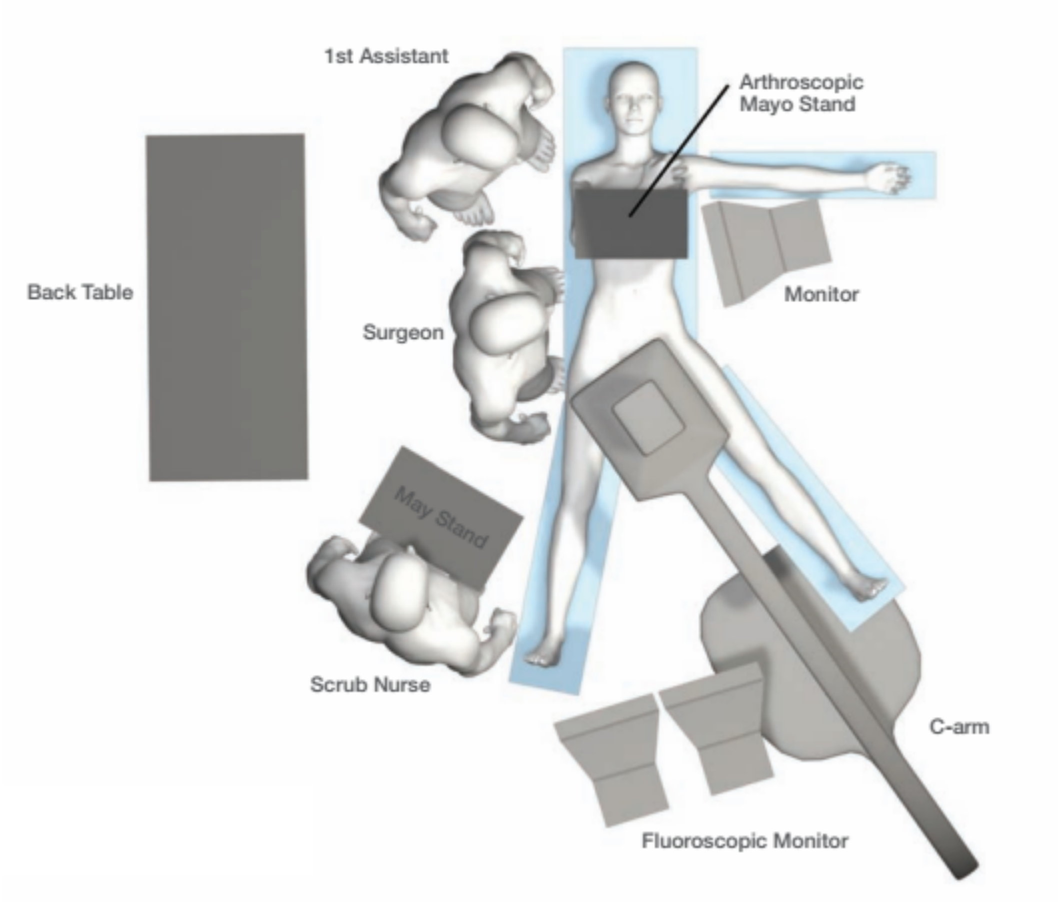


Figure 5. Typical HA operating room set-up [1]

A common technique today is the use of a foot plate in order to dislocate the hip for the HA procedure. A footplate is a flat plate device with handles that an operator will use to turn the leg and distract the hip (Figure 6). For this technique, the majority of the lower leg is wrapped and attached to the footplate via wrapping [7]. One major benefit of this method is it can be utilized with various hip fracture tables (HFTs), therefore eliminating the cost of a new table. During this operation, feet, shins, and the perineal post are well padded. The non-operative foot is placed in an HFT clamp. The operative hip foot is placed over a mayo table that is padded with blankets or a pillow. An operator will turn the operative foot while a surgical assistant positions the patient's hip. This method is useful because it can be performed with a variety of distractor devices that are currently in use. However, this method does require a trained surgeon and a surgical assistant to be present in the operating room [7].

Figure 6. Foot placement for Mayo Table HA method [9]

In order to prevent heel lift and the heel slipping out of the boot, Arthrex designed and made a boot that safely avoids this issue. The Arthrex table is capable of setting up the entire procedure which includes the boots themselves (Figure 7). The novel traction boots consist of a ratcheting mechanism that limits the movement of the heel. There are a couple major downsides to this mechanism including its large cost and the amount of space it takes up in the operating room [8].

Figure 7. Arthrex hip distraction method [8]

Stryker has developed its own method for hip distraction that is both cost effective and space efficient. Stryker utilizes a boot to secure the operative side foot and a pulling mechanism to distract the hip (Figure 8). The pulling mechanism is secured to the operating table prior to patient arrival.

Figure 8. Stryker hip distraction mechanism

First, a soft foam foot pad is secured around the foot using Velcro straps. Then the foot is placed in the bottom portion of the boot. Then the surgeon or nurse will secure the straps on the top portion of the boot to the latches on the bottom portion. Once the straps are in place, a twisting mechanism is used to tighten the boot. The surgeon or nurse is instructed to turn the three knobs on the top of the boot using their thumb and pinky finger until they can no longer turn (Figure 9). This parameter varies between each person who tightens the boot. Next, the boot will be secured using a pin into the distraction device, and the surgeon will be ready to distract the hip.

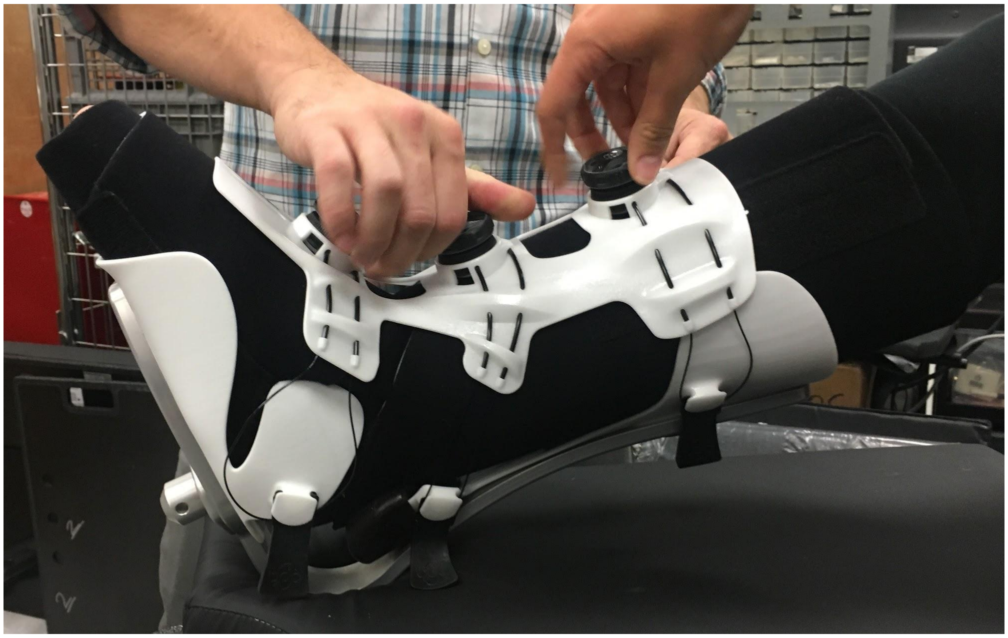


Figure 9. Tightening of boot on patient

With any of these hip distraction methods, two main issues arise; nerve damage and heel slipping within the boot. Many patients who undergo HA experience numbness in their limbs following surgery. This issue is due to lack of blood flow and nerve pinching caused by the distraction device. During the operation, there is also a large risk of the heel beginning to slip, therefore narrowing the space available for operation, and possibly compromising the entire procedure. Currently there is no method to monitor both nerve damage and heel slip within available distraction methods. The addition of sensors to detect these issues during operation could greatly improve Hip Arthroscopy for both doctors and patients.

When the body, more specifically the ankle/foot, is exposed to tight compression, patients will experience deficient blood flow and become at risk of avoidable injuries [9]. During HA, when the patient is subjected to anesthesia, he or she will be unable to identify if the boot is on too tight, resulting in loss of blood flow. By experiencing a loss of blood flow in the foot, the patient will battle vascular obstruction of all 3 major arteries at the ankle level as well as a diminished sensation of the superficial peroneal nerve. Damage to the superficial peroneal nerve can be very severe and result in the inability to turn the foot laterally [10]. In order to avoid these consequences, the boot must be heavily padded, especially around the ankle area. This allows for extra cushion around the nerves and increases the surface area due to the boot buckles applying a high compression load. By increasing the surface area, the patient experiences a distributed load rather than a point load, which reduces the risk of nerve damage. Avoiding this issue may be difficult, but currently, there are no practical methods for doctors to detect the lack of blood flow during the HA procedure, resulting in the demand for innovative method.

Although the heel slipping causes no immediate damage to the patient, it increases complications for the surgeon immensely. When the foot is placed inside of the boot there is no way to secure the foot against the sole outside of the latching mechanisms around the shin. Once the patient is placed at an angle and the doctor starts to work on the hip, it becomes possible for the hip to shift due to the natural forces of the muscles attempting to correct the joint into its natural position (Figure 10). After heel slip occurs during surgery, the surgeon will have limited space because the femoral head will begin to shift back into its natural position and the success of the surgery will decrease. Doctors are put in a critical situation once the heel slips because it is difficult to reset the hip. If the surgeon recognizes the heel slip and determines that the distraction gap has become too small, then the hip most be distracted out of position again. If the space becomes too limited from the slip, doctors may be forced to stop the surgery and repeat the entire process of distraction. Finding the middle ground between making the boot too tight and being able to keep the foot in a consistent location is crucial for a successful hip arthroscopy.

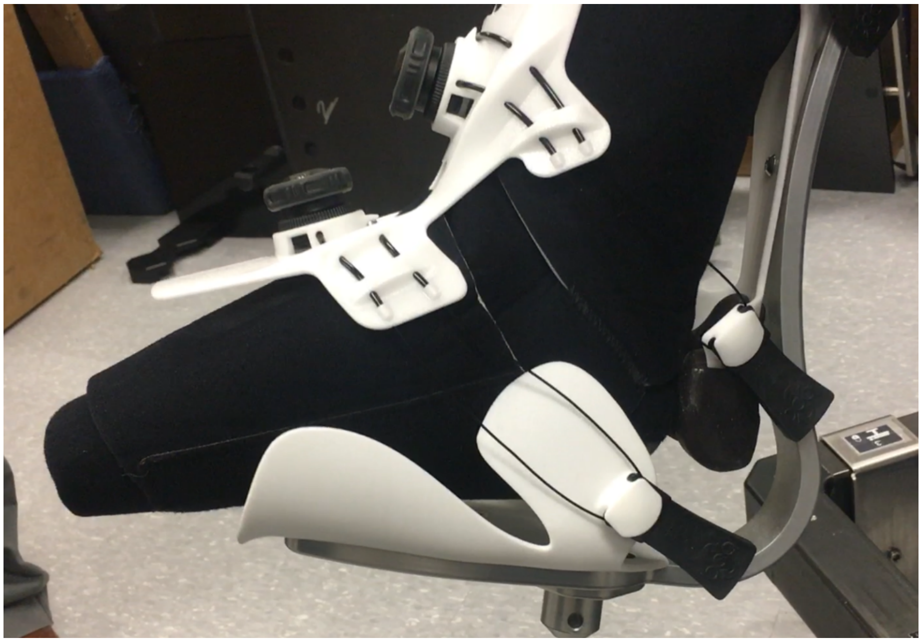


Figure 10. Stryker hip distraction boot after heel slip

In order to guide the design process, two quality regulation standards will be used. The U.S. Food and Drug Administration standard, 21 CFR 820 Quality System Regulation for Medical Devices regulates, "the design, manufacture, packaging, labeling, storage, installation, and servicing of all finished devices intended for human use" [11]. ISO 13485 is the Quality Systems Management standard for medical devices set by the International Organization for Standardization. The standard "sets out the requirements for a quality management system specific to the medical devices industry" [12]. Both quality standards are general in scope to all medical devices. The project sponsor, Ryan Dahlby, informed the team that Stryker's internal standards are largely determined by codes and standards from external agencies. As a result, teams at Stryker rarely refer to external standards documents to guide their product development. PolyHIPS will use the FDA and ISO standards documents to guide their design process and supplement regulatory guidance from the team's sponsors.

Hip Arthroscopy is a fairly new procedure, and the ability to monitor nerve damage and heel displacement will drastically improve the situation of patients and doctors. The PolyHIPS team will design and prototype a hip arthroscopy boot that identifies principal operator concerns such as heel slipping and nerve damage.

Chapter 3. Design Development

1. Nerve Damage

Currently, most methods of nerve signaling require user feedback, electrical stimulation and/or electrical recording of the brain. Because hip arthroscopy requires patients to be unconscious and mechanical tools should not be electrically stimulated, current technology for detection of nerve damage is very limited. An alternative, direct detection method was to examine blood flow due to the fact that nerve bundles are tied closely to blood vessels.

After investigation of the various ways to detect nerve damage and blood flow, it was determined that the following were viable technologies; pulse oximetry, laser doppler, temperature sensing, ankle brachial index, nerve signaling, and pressure sensors. Pugh matrices were formed depending on the intended target of detection; one for blood flow and one for nerve damage, refer to Appendix B. These matrices determined nerve signaling did not satisfy the safety requirements, but the other five technologies were analyzed and tested.

a. Pulse Oximetry

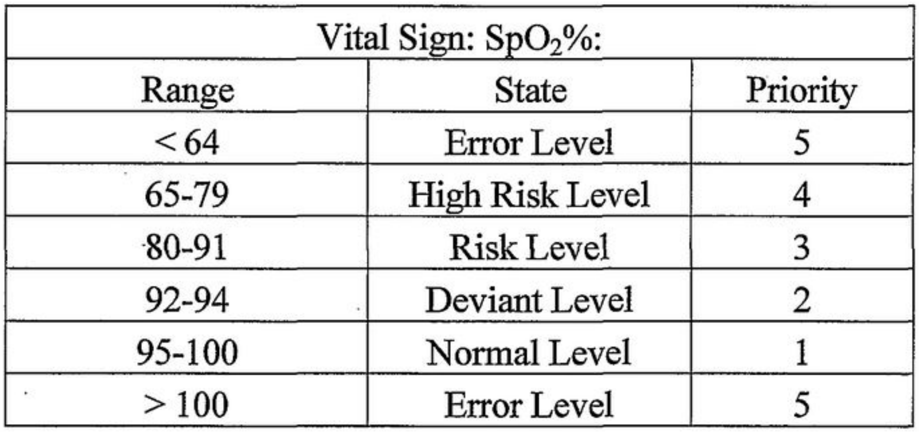
One of the most common methods of monitoring blood flow during a procedure is the use of a pulse oximeter (Figure 11). This non-invasive device works by sending a light through the probing zone of the device and using the amount of light absorbed by the finger to calculate the oxygen levels in the blood. Based on the concentration of light absorbed and the distance traveled through the pulse oximeter, the amount of oxygen being carried in the hemoglobin can be identified [13].

Figure 11. Image of a pulse oximeter reading the vitals through the finger [14]

This device is commonly used on patients prior to the surgery to understand the patient's vitals; allowing the surgeon to constantly understand the condition of the patient. Being able to access this information would limit the possibility of neuropraxia and improve patient safety during the surgery, ultimately improving their experience.

i. Oxygen Saturation

Pulse oximeters are constantly measuring the oxygen levels in the blood, and this data can be transmitted to an external device for the surgeon or nurse’s reference. The oxygen saturation (SpO2) measurement on a pulse oximeter is the amount of oxygenated hemoglobin as compared to non-oxygenated hemoglobin in the blood [15]. The normal oxygen saturation levels range from 95-100% [16]. Expected physiological abnormal parameter values were obtained, shown in Figure 12 [17]. This measurement might not directly measure blood flow because blood that is static in the toe might still have ample oxygen, which would give a normal SpO2 reading. As a result, a SpO2 reading might not give a measurement that represents circulation, or possible nerve damage.



Vital Sign: SpO_2%:		
Range	State	Priority
< 64	Error Level	5
65-79	High Risk Level	4
80-91	Risk Level	3
92-94	Deviant Level	2
95-100	Normal Level	1
> 100	Error Level	5

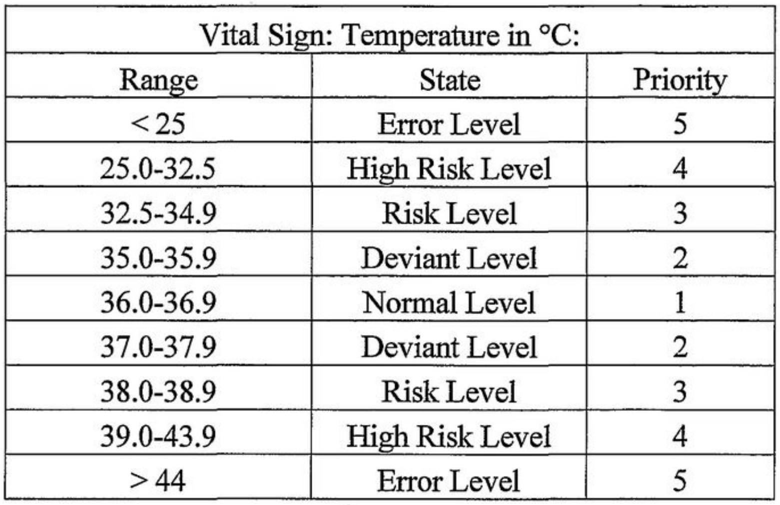
Figure 12. SpO2 levels and the relative severity of the deviation from normal.

ii. Perfusion Index

A pulse oximeter also displays perfusion index, which is the ratio of pulsatile to nonpulsatile blood flow. Perfusion index ranges from 0.02% to 20%, where 0.02% shows extremely low perfusion rates and 20% is an extremely high perfusion rate [18]. Currently, surgeons pinch the patient’s toe and visually track the amount of time it takes for the toe to turn pink again during hip arthroscopy. This allows surgeons to have a general sense for the blood flow within the foot when the boot is tightened. The amount of time it takes for color to return to tissue after pressure has been applied to the area is capillary refill time [19]. Perfusion index is directly correlated to capillary refill time, so the perfusion index measurement allows surgeons to gather more precise readings of blood flow to the foot [20]. As compared to oxygen saturation, perfusion index gives a better reading for local blood flow in the foot.

b. Temperature Sensing

Temperature sensing is an indirect method to detect lack of blood flow. The response of skin blood flow to a change in skin temperature is contingent upon rate-dependent memory and can be highly variable [21]. However, the general trend indicates less blood flow will lead to lower skin temperature. The mean toe temperature of adult regardless of foot movement or sex is 34.0˚ +/- 1.8 ˚C [22]. Normal and abnormal temperature ranges were obtained to determine the severity of the lack of blood flow, shown in Figure 13 [17]. It is expected that the toe temperature decreases as the amount of blood flow decreases.



Vital Sign: Temperature in °C:		
Range	State	Priority
< 25	Error Level	5
25.0-32.5	High Risk Level	4
32.5-34.9	Risk Level	3
35.0-35.9	Deviant Level	2
36.0-36.9	Normal Level	1
37.0-37.9	Deviant Level	2
38.0-38.9	Risk Level	3
39.0-43.9	High Risk Level	4
> 44	Error Level	5

Figure 13. Normal and abnormal temperature ranges for distal phalanges.

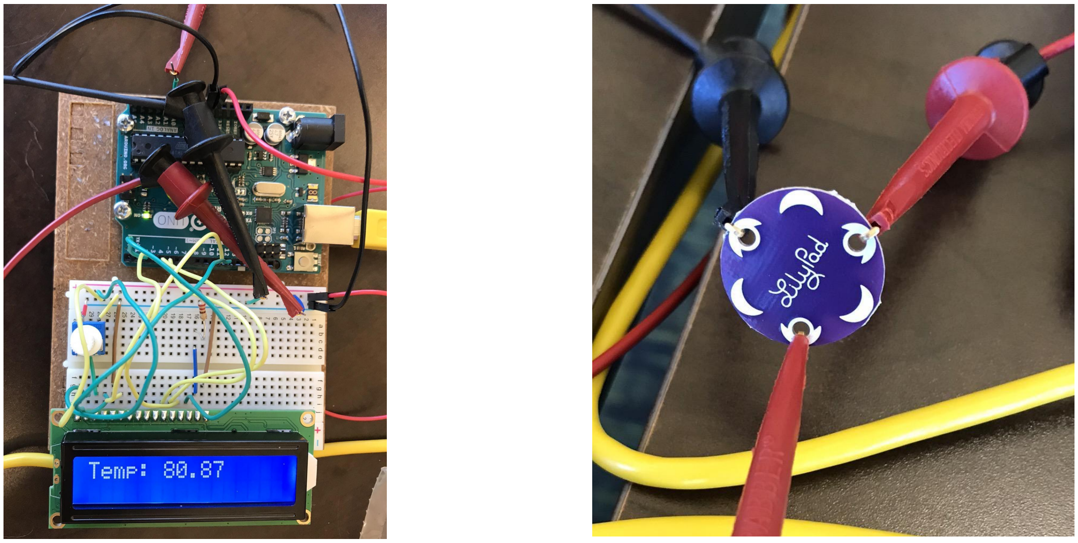
i. FLIR ONE Thermal Imaging

FLIR ONE is a thermal imaging camera attachment for a mobile phone. The device converts infrared energy (heat) into an electrical signal, which is then processed to a color image. The camera was able to detect normal foot temperatures (Figure 14).

Figure 14. FLIR ONE thermal image of right foot

ii. LilyPad Temperature Sensor

The MCP9700 is a small thermistor type temperature sensor. The sensor was connected to an Arduino Uno and display LED (Figure 15). The sensor recorded ambient room temperature at 22 ˚C and a forefinger temperature at 36 ˚C. A wrap was placed around the proximal phalange to emulate lack of blood flow. The temperature decreased to 29 ˚C.



A. **B.**

Figure 15. A. Temperature display and Arduino breadboard B. Temperature sensor attached

Temperature gradient of the toe is used as an indication for success of surgery. In one experiment, patients with a large temperature gradient during and post-surgery had a higher indication for successful recovery. [23]. Consistently, temperature sensing remains highly variable; it depends on the patient, operating room, and other factors. Additionally, the gradient between during and after surgery would not be obtained because the boot and apparatus is removed post-surgery. Furthermore, most scientific literature for temperature sensing were for normal subjects during steady states. There is no adequate data currently available relating temperature to blood flow for abnormal conditions. Because of these factors, there is not much application for temperature sensors for the device and it was eliminated as a viable sensor for blood flow.

c. Ankle Brachial Index

At present, ankle brachial index (ABI) is used to diagnose peripheral artery disease because it determines the level of blockage in arteries. It's a ratio of systolic blood pressure of ankles over arms and a value below 0.9 can indicate a potential problem (Figure 16).

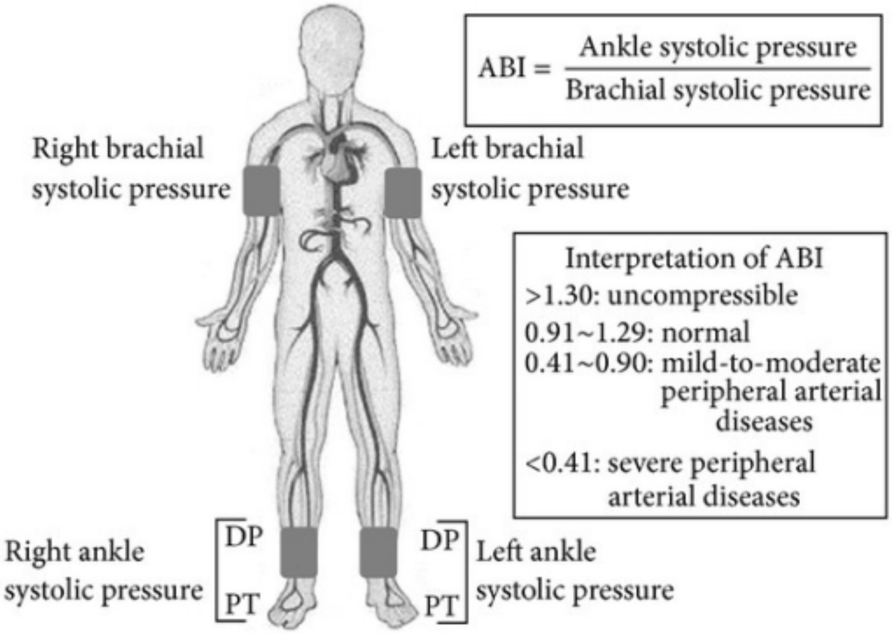


Figure 16. Anatomical placement of pressure cuffs and interpretation of ABI values [24]

ABI is a direct method for determining the level of blood constriction or the lack of blood flow the patient experiences. The following are the procedural steps for taking ankle brachial index: 1. Patient must lay down flat in face-up position for 10 minutes to normalize. 2. Brachial pressure is obtained by attaching a pressure cuff above the antecubital fossa, aka the middle part of the elbow bend. 3. Ankle pressure is obtained by attaching a pressure cuff two inches above the malleolus, the bony knob, of the ankle. 4. Divide ankle systolic pressure over brachial systolic pressure to receive the ABI ratio.

This method satisfies almost all of the customer and engineering requirements, but it might be too bulky for the boot design and it could increase the operating room setup time slightly. The wires and external hardware for the blood pressure cuffs might take up excessive space, and more importantly, the increased pressure surrounding the pressure cuff from the boot could lead to inaccurate readings. The team performed some tests in order to verify the accuracy of ABI within the boot.

d. Nerve Conduction Velocity (NCV) Test

Pulse oximetry, temperature, and ABI are all methods that can help quantify blood flow and condition within the foot. These are indirect methods of measuring nerve damage and can help determine compression. However, a more direct approach to detecting nerve compression was sought after.

At first glance, it would make the most sense to directly measure nerve signaling at the compressed site of the foot. This would provide a direct reading as to whether or not the nerve is signaling correctly. A noninvasive technique should be used, or else this would overcomplicate the procedure. The most common way to do this is using a Nerve Conduction Velocity Test, which measures how fast and strong the electrical activity is in a nerve [25]. This test utilizes electromyography (EMG) signaling. In this test, a nerve stimulator targets a nerve to cause a muscle contraction. Surface electrodes will be placed on muscles of concern that are stimulated. The electrodes will provide a readout of the electrical signal received by the muscle and can therefore determine if the muscle is functioning properly.

Ultimately, this detection of nerve signaling proved to be too complicated for the Stryker boot application due to several concerns. First, the location of the targeted nerve varies greatly from patient to patient. The hospital staff would have to take time to find and stimulate the desired nerve before placing the boot on. Also, each patient will have a different nerve impulse threshold, and some patients may require a much higher stimulus to create a muscle contraction. This will add additional time to the procedure for the hospital staff to determine the patient's nerve impulse threshold. Most importantly, this method imposes an additional electrical current into the operating room environment, which will require grounding, and create a greater safety hazard. For all of these reasons, Nerve Conduction Velocity was eliminated as a nerve function detection method.

e. Pressure Sensors

Currently, pressure on the foot when tightening the boot varies by patient. This is because surgeons use the maximum possible torque from their thumb and pinky finger to determine the tightness of the boot. Pressure sensors can help quantify the amount of force the physician applies and standardize the boot tightening procedure. Furthermore, the pressure applied to nerve bundles in the ankle and foot will correspond to the resulting nerve damage in the foot after surgery. By monitoring the amount of pressure that the boot is exerting on the foot, surgeons will be able to prevent the boot from being tightened to levels of pressure that could be harmful to the nerves in the foot. Pressure Sensors provided a safer alternative to determine nerve compression that could lead to damage. The Adafruit Round Force Sensitive Resistor (FSR) was tested to quantify compressive forces on nerves during HA. Force sensitive resistors detect pressures by changing its resistive value depending on the force they experience. The Interlink 402 Round Force Sensitive Resistor has a ½” diameter active area with a force range of 0 to 20 lbs (Figure 17). The sensor detects varying pressure levels when wired to the Arduino.

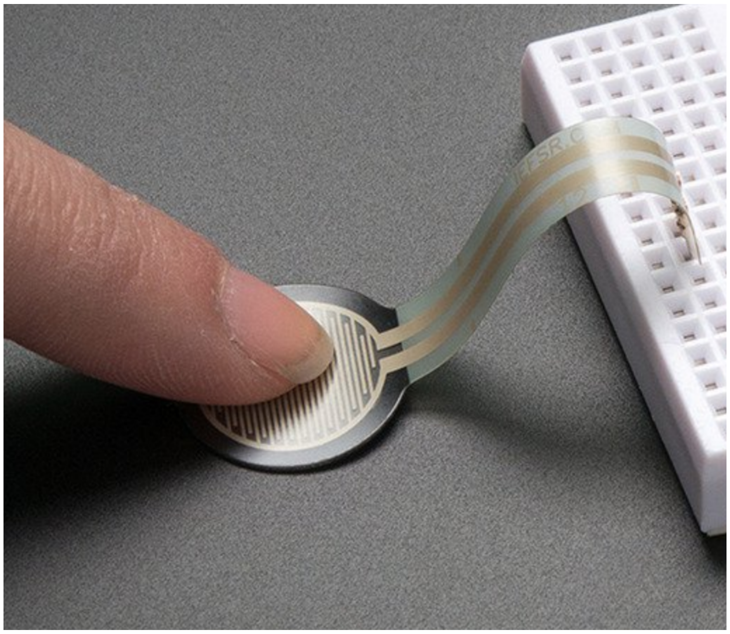


Figure 17. Force sensitive resistor, Interlink 402 [26]

f. Vascular Doppler

As shown in Figure 18, a vascular doppler directly measures blood flow by using high-frequency sound waves [27]. The device uses ultrasound to gather systolic and diastolic blood pressure, pulse rate, and arterial blood flow velocity [28]. Vascular dopplers are commonly used to detect poor circulation, particularly for patients with peripheral artery disease and monitoring lower limb perfusion when a tourniquet is used. This device can be used all over the body but are most commonly used on the lower limbs.

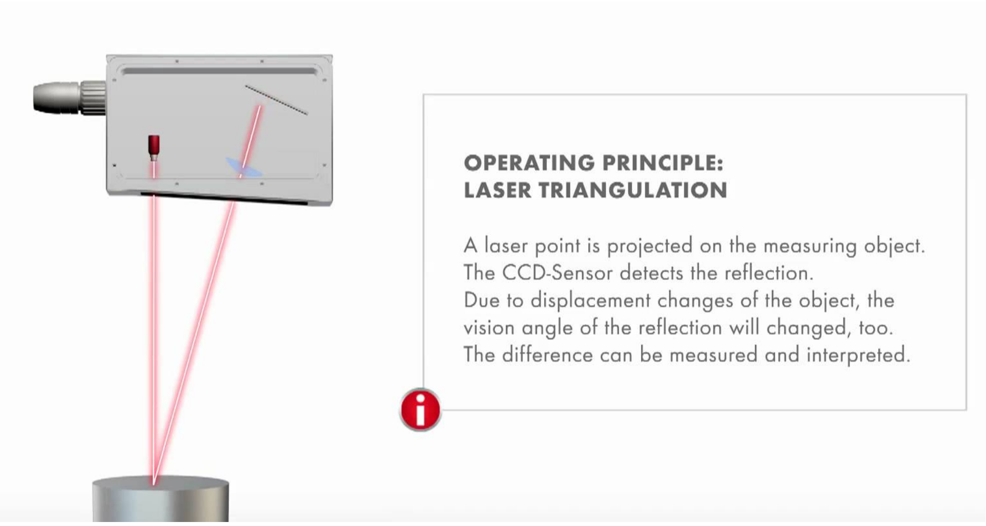
Figure 18. Hadeco Bidop Bi-Direction Waveform LCD Vascular Doppler [29]

Vascular dopplers can be used with various probes, such as vascular probes, temperature probes, probes with pressure cuffs for the toe, and Photoplethysmography (PPG) probes. For this application, a PPG probe was used instead of the Doppler probe to eliminate some of the noise in the data provided by the Doppler probe. A PPG probe is the same light-emitting probe that is used on a pulse oximeter, but it is more sensitive when it is used with the vascular Doppler. A toe clip can also be used to hold the probe in place to provide a hands-free configuration.

2. Heel Slip

In order to detect whether the heel is slipping, three different concepts were considered. A method to constantly detect how far the heel has slipped is required, however several different approaches are available. The most obvious method to detect heel slip is to mount a camera which faces the transverse direction of the foot. This camera would be hooked up to a monitor which would allow the nurse/surgeon to constantly check the location of the heel. This method is very beneficial because it gives a live image feed of where the heel is, however, the exact distance from the heel to the sole of the boot would be unknown. Another flaw with this idea is that someone would have to constantly check the live feed in order for this method to be useful. There would be no alarm or alert that would inform the surgeon that the heel has begun to slip. Nonetheless, this method has numerous advantages over the current method surgeons, which consists of the nurse sticking his/her finger in the gap between the heel and the boot to confirm the distance. Having an autonomous device to monitor and alert heel slip would be much more beneficial for this type of operation.

A common method of measuring movement is using a laser displacement sensor. A laser displacement sensor is a gauge to measure how much an object moves relative to a set reference point. Displacement is measured by using lasers to send a signal and have it ricochet off the object and return back to the receiver in the sensor. The sensor then sends the time it takes to receive the laser and converts that time into a distance. The sensor is constantly reading the laser, therefore, when the object moves, it can instantly be transmitted to the desired monitor (Figure 19). Having a method like such would allow for a surgeon to focus primarily on the surgery, and only worry about the heel slipping if it detects a distance deemed dangerous. This distance is when the heel begins to cross the calcaneus pad on the boot, which is located a few inches off of the sole of the boot. Testing was done with this sensor to determine the range of the sensor and the accuracy of it, both of which performed as expected.



OPERATING PRINCIPLE:
LASER TRIANGULATION

A laser point is projected on the measuring object.
The CCD-Sensor detects the reflection.
Due to displacement changes of the object, the
vision angle of the reflection will changed, too.
The difference can be measured and interpreted.

Figure 19. Laser Triangulation Method used in Displacement Sensors [30]

Along with the laser displacement sensor, there are other methods of measuring displacement such as a spring-loaded displacement sensor. The way this sensor works is by having a probe attached to a spring, and when the spring expands or contracts, the sensor is triggered, and the displacement can be found depending on the controller/transmitter being used (Figure 20). This sensor would raise concerns in regard to how it would be integrated to the boot, and whether that would affect its performance. The other flaw with this sensor is that it could easily be damaged by the force of the heel on the boot as these sensors are typically very sensitive and must be placed in a secure location.

Figure 20. Spring Loaded Displacement Sensor to measure movement [31]

After constructing the Pugh Matrix for the heel slip issue in Appendix B, the laser distance sensor ended up being the best choice. Due to the fact that the sensor must be able to wirelessly connect to an Arduino, the spring-loaded sensor and camera are ruled out. Likewise, the spring-loaded sensor would require extra work for the surgeon to attach the sensor to the heel of the patient, causing a change in procedure. Although the spring-loaded sensor and laser displacement sensor are similar in price, the camera method would be too costly, ruling that option out entirely. Finally, the spring-loaded sensor is not intuitive and would not have a logical integration plan to fit into the boot, therefore the laser distance sensor is the optimal decision.

3. Software and Connection

From the sensors, the data must be collected, managed, and wirelessly connected to a tablet. In order for the data to be transmitted, it must first be connected to a microcontroller. The sensors are wired to a local microcontroller on the boot. The prototype for this design wired the sensors up the metal spine, as shown in Figure 21.

Figure 21. Wiring Configuration

The microcontroller serves as both a hub for data as well as the power supply for the sensors. The Arduino Uno, shown in Figure 22, is the current model used for prototyping and testing.

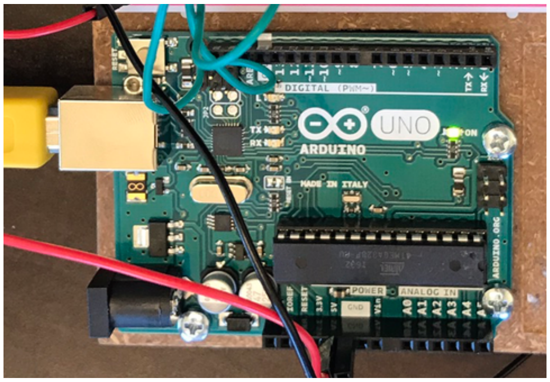


Figure 22. Arduino Uno Microcontroller

Additional components are required to transmit the data from the microcontroller to the tablet. Options for wireless data transfer include XBee modules, Bluetooth, and Wi-Fi. XBee modules are small, wireless connectors that transmit small amounts of data and have low power consumption [32]. Wi-Fi data transmission requires access to the internet and a router. Bluetooth technology allows wireless connection between devices [33]. Bluetooth and XBee modules are more favorable because they do not require internet connection to transmit data. XBee modules have lower data rates and lower power consumption compared to traditional Bluetooth technology [34].

Bluetooth Low Energy (BLE) is an alternative that uses Bluetooth technology, but with lower power consumption [35].

Chapter 4. Preliminary Testing

An outline and timeline of the preliminary tests for the development of the final design can be found in Appendix G. The Design Verification Plan and Report was used for both preliminary testing and testing the final design.

1. Nerve Damage

a. Pulse Oximeter Testing

i. Acc U Rate Pro Series 500 DL

Acc U Rate Pro Series 500 DL was used as a method to determine if a standard pulse oximeter could be used on an adult toe. It was able to detect saturation levels on the toe intermittently, but it did not give a consistent reading (Figure 23). When the boot was tightened, the device did not read saturation levels at all.

Figure 23. Pulse oximetry of the toe when boot is attached

There were multiple limitations listed on the manufacturer's website for the Acc U Rate Pro Series 500 DL such as pulse rate range (30-240 bpm), pulse-filling rate (0.2%), and accuracy deviation (+/-2%). Additionally, a large hindrance is that most pulse oximeters are designed for the finger, so the standard pulse oximeter does not fit on an adult toe.

ii. Masimo RAD-57

The restrictions of the Acc U Rate Pro Series 500 DL used for testing were mitigated by purchasing a more accurate pulse oximeter with a disposable adhesive wrap (Figure 24). Masimo's RAD-57 pulse oximeter was evaluated with their LNCS Neo-3 adhesive wrap sensor. This sensor is commonly used for neonatal applications, but it can also be used to monitor adult saturation levels. The adhesive wrap can be easily wrapped around a variation of toe sizes and is disposable after each use.

Figure 24. Masimo RAD-57 Test Setup with LNCS Neo-3 Adhesive Wrap

Preliminary testing of this device showed concerning results (Figure 25). The torque on the BOAs was increased from 0 to 5 in-lb every four minutes. The perfusion index (PI), or the ratio of pulsatile to non-pulsatile blood flow, was measured every minute. These values should be decreasing as the boot is tightened and as the boot constricts the foot over time because blood flow is being cut off by the pressure of the boot. Furthermore, the pulse oximeter completely lost the signal at 11 minutes, 14 minutes, and 15 minutes.

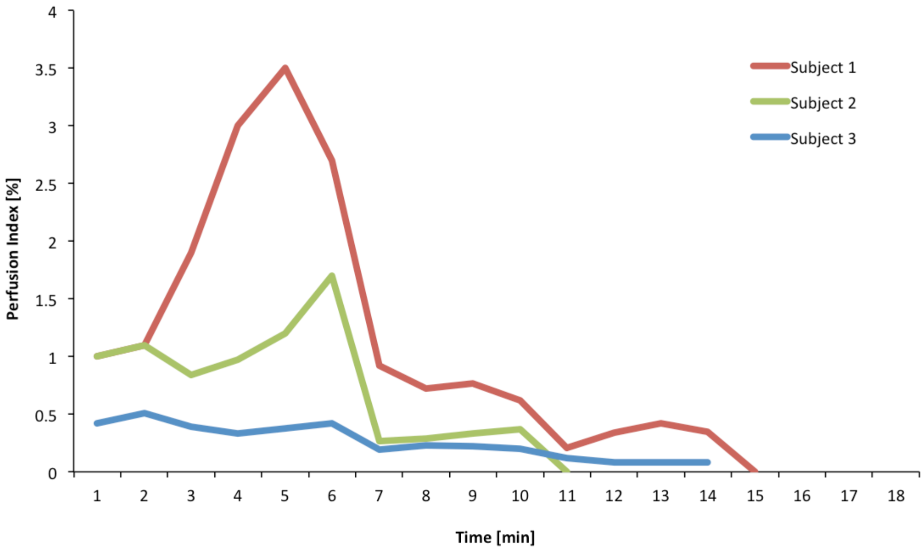


Figure 25. Perfusion Index Monitored Over Time

Because the perfusion index measurements did not display a change that directly correlated with boot tightness and the device experienced signal loss, it was concluded that this method would not be sufficient for tracking nerve damage caused by the boot.

b. Vascular Doppler

The preliminary test protocol for the Koven Bidop 3 closely resembled the protocol for pulse oximetry testing. The PPG toe clip was attached to the patient's grand toe and the peak-to-peak measurement of the blood flow waveform was read from the screen (Figure 26).

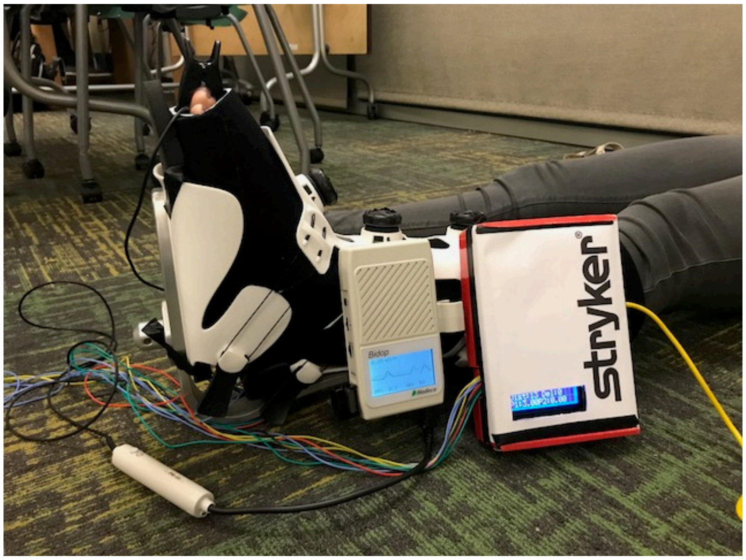


Figure 26. Koven Bidop 3 Test Setup with PPG probe and Toe Clip

This configuration was tested on five subjects over multiple days. The peak-to-peak values were recorded over a range of 0 to 5 in-lb applied to the BOAs. Figure 27 shows an example of one day of preliminary testing on the vascular Doppler.

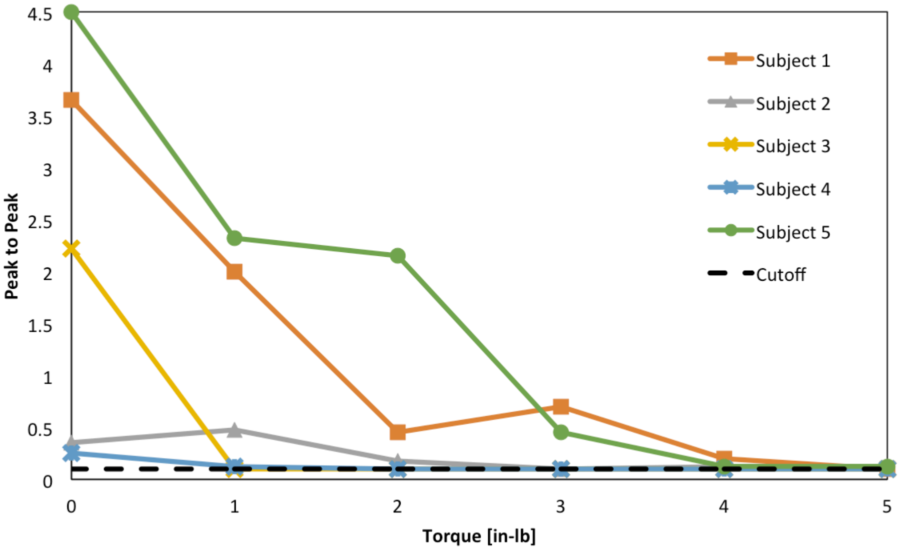


Figure 27. Peak to Peak Blood Flow versus Torque Applied on BOAs

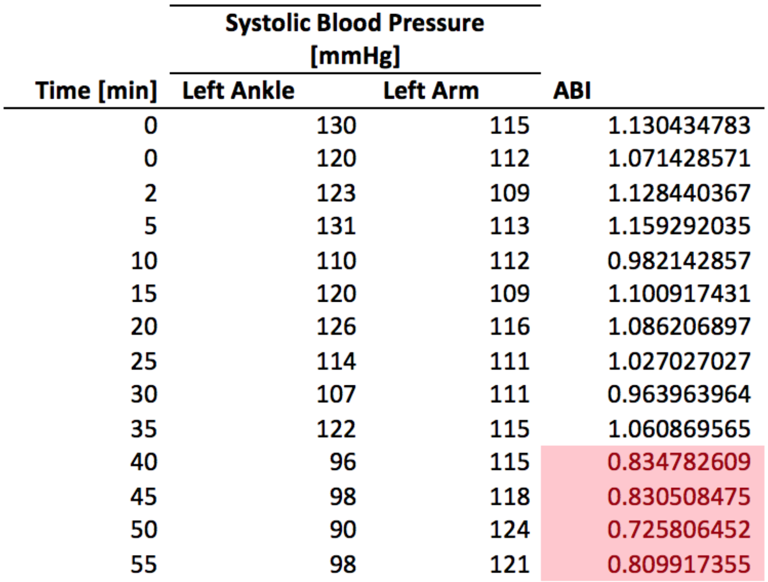
Figure 27 shows a more promising overall trend because the peak-to-peak blood flow values decrease as torque increases. Subjects 1 and 5 show more consistent decreasing trends that level off at 4 and 5 in-lb. Subject 4 shows a relatively flat line that stays consistently at 0.1. While there was no signal loss in any of the tests, the tendency for the peak to peak values to level off at 0.1 is concerning because this is the lowest value that can be read before the Doppler hits 0. This technology could potentially work for patients that have perfusions rates similar to Subjects 1 and 5, but it is not feasible for the other three subjects. Since the boot will be tightened to at least 3 in-lb during surgery, it was determined that this technology is not sensitive enough and the peak to peak measurements will likely stay leveled off at 0.1 throughout surgery. The leveled off value does not tell the surgeon any relevant data as the measurement is essentially at 0.

Although this technology was not sensitive enough to monitor blood flow in this application, the preliminary testing provided important information regarding the blood flow in the foot during this procedure. The data showed that there is very little blood flow in the foot during the procedure. Although this may seem concerning, other devices used in surgical procedures, such as tourniquets, do not allow blood flow to the lower limbs. This led the team to conclude that while the low perfusion rates are acceptable, the blood flow could not be tracked with devices the currently exist on the market.

c. Ankle Brachial Index

Testing was performed to assess the sensitivity of the blood pressure cuff under the soft foam foot pad. A blood pressure cuff was positioned two inches above the malleolus of the left foot and another cuff on placed just above the antecubital fossa of the left arm. The boot was tightened using the thumb-pinky method. Every 5 minutes, systolic blood pressure was taken for a total of 1 hour. At time 40, the ABI dropped below the normal levels (0.91-1.29); this increases the viability of this method (Table 2).

Table 2. ABI with Boot Secured for 60 Minutes



	Systolic Blood Pressure [mmHg]		
Time [min]	Left Ankle	Left Arm	ABI
0	130	115	1.130434783
0	120	112	1.071428571
2	123	109	1.128440367
5	131	113	1.159292035
10	110	112	0.982142857
15	120	109	1.100917431
20	126	116	1.086206897
25	114	111	1.027027027
30	107	111	0.963963964
35	122	115	1.060869565
40	96	115	0.834782609
45	98	118	0.830508475
50	90	124	0.725806452
55	98	121	0.809917355

Another test was conducted to examine the effects of external pressure on the pressure cuff. A blood pressure cuff was once again placed just above the antecubital fossa of the left arm and 3 control readings were recorded. After 3 minutes, an external force was applied by a secondary person wrapping their hands around the cuff and introducing a pressure. Three additional readings were recorded.

These results were inconsistent with control values and the testing revealed a pressure tolerance that was considered abnormal when the patient themselves were perfectly fine. The main concern for this technology was decreasing patient safety because a pressure was being applied in addition to the pressure the boot currently places on the patient. The team justified its risk because it was a direct measurement of blood flow, but as data suggests, ABI was inaccurate. This decreases the validity of using ABI for this application and eliminated this technology from our final design.

d. Pressure Sensors

When wired to the Arduino Uno, the pressure sensors provide a Force readout in Newtons. The pressure sensors were connected to two lengthening wires to allow them to be placed within the boot while it was tightened. Two pressure sensors were successfully wired to the Arduino Uno and simultaneously output Force values. These pressure sensors were tested on the black calcaneal hold within the boot since this is suspected to create a large compressive force on the posterior portion of the ankle (Figure 28).

Figure 28. Pressure Sensor Placement during Preliminary Testing

The pressure sensor output a value from 0 N to 13 N. At typical surgical tightness (as tight as the thumb and pinky will allow) the pressure sensor read out 8 N. 13 N was read when the boot was tightened to maximum allowable on the foot, which would exceed that experienced during any procedure. This force range was under the 20 N maximum for the selected FSR pressure sensor. The surgical force readout and max force readout were converted to pressure in mmHg based off the area of the sensor as specified in equation 1.

$$Pressure \ = Force \ / Area \ of \ Sensor \ [1]$$

This yielded the following corresponding values of pressure shown in Table 3.

Table 3. Pressure Sensor Output

Force [N]	Pressure [mmHg]
8	12.03
13	19.55

This preliminary testing proved that the FSR pressure sensors were able to output a pressure value when placed within the boot at full tightness, and that the maximum Force experienced did not exceed the spec limit of the sensor.

Testing was preformed using a blood pressure cuff to ensure that the sensors increase with an increase in applied pressure. Two pressure sensors were placed under the cuff (Figure 29).

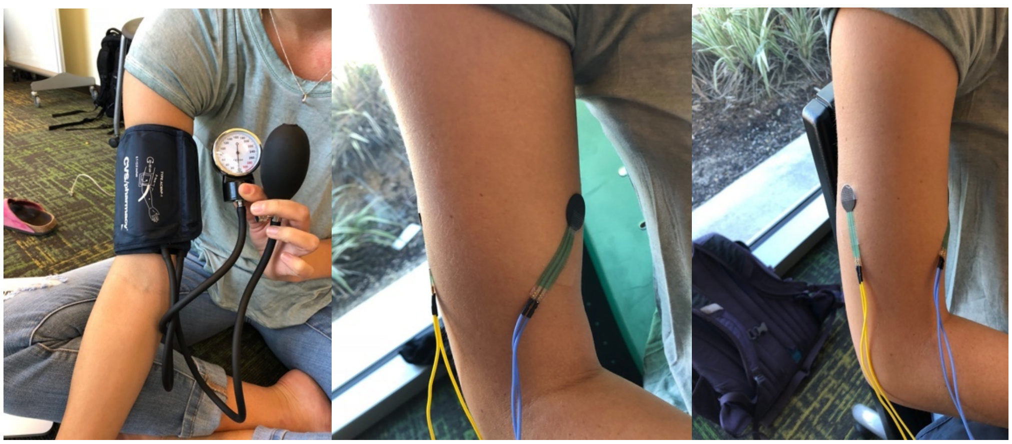


Figure 29. Pressure sensor placement under blood pressure cuff.

The pressure cuff was incremented from 0 to 100 mmHg by increments of 20 mmHg and 3 force readings were taken at each increment. Average force reading for each sensor at each increment are plotted next to expected baseline values (Figure 30).

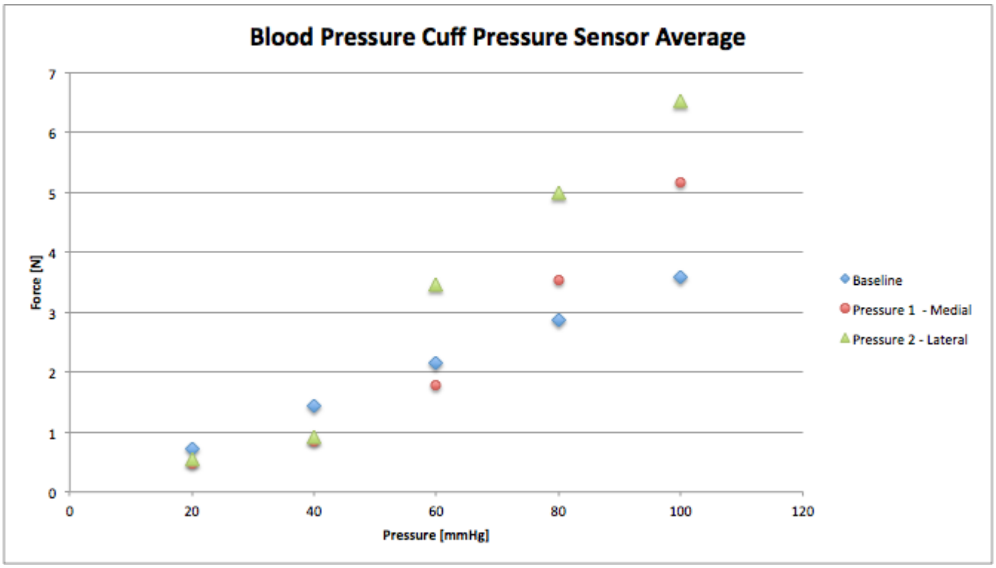


Figure 30. Blood pressure cuff pressure sensor reading

This experiment proved that the pressure sensor readings increase incrementally with externally applied pressure. Differences in pressure readings between the two pressure sensors was attributed to location on the skin. The medial pressure sensor was placed on softer skin (had more give) would could have led to the decreased pressure values recorded. Moving forward, the surface in which the sensor was being placed on was taken into account.

Similar testing was performed on the calcaneus to ensure pressure would increase on the surface of the heel during surgical configuration. Pressure sensors were placed on both the medial and lateral side of the ankle (Figure 31).

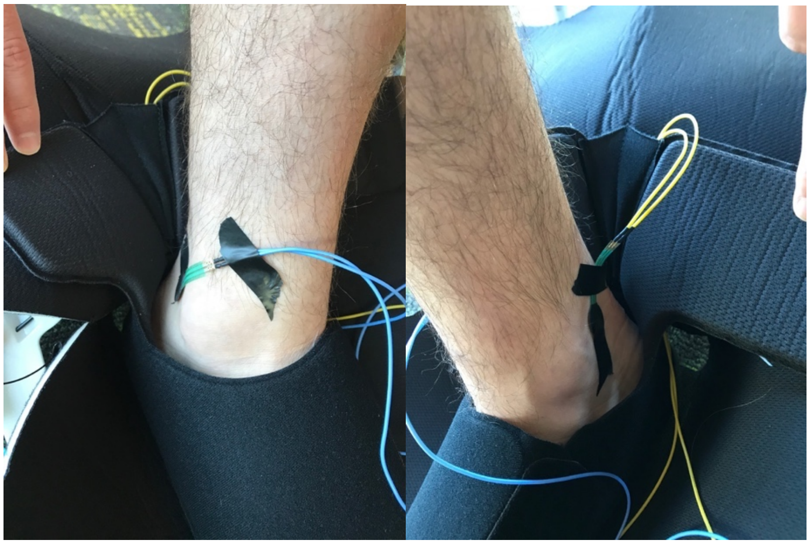


Figure 31. Calcaneus pressure sensor placement.

Testing was done prior to acquiring a torque gage, so a range of 5 to 6.5 turns was tested in increments of half a turn. Pressures were also recorded when releasing each BOA clip to ensure a pressure drop was present (Figure 32).

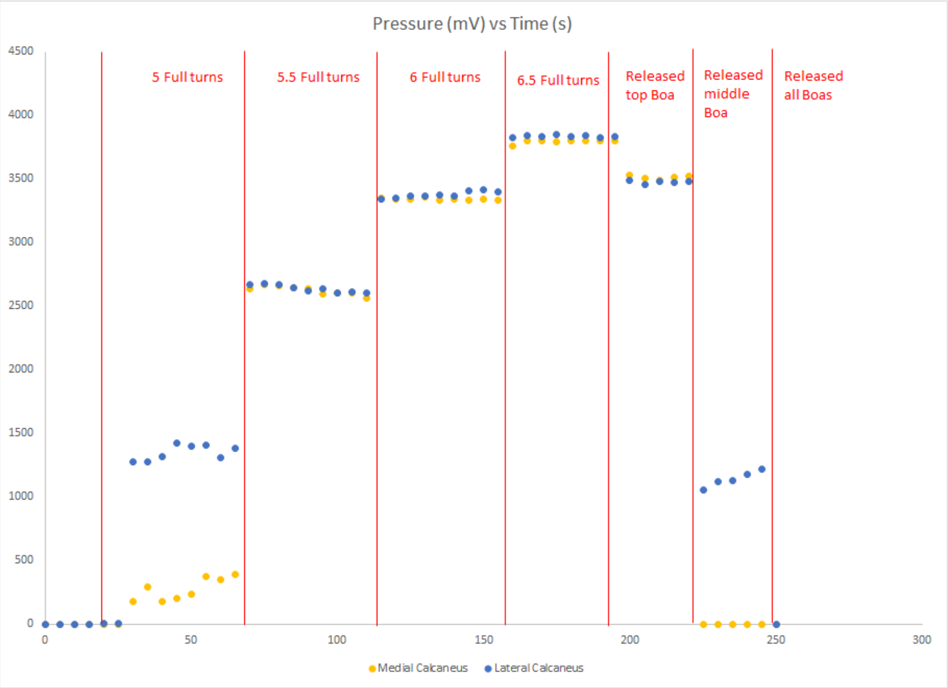


Figure 32. Calcaneus pressure sensor readings with increasing BOA tightness.

Results indicated that the pressure sensors showed the same incremental increase while the boot was tightened on both sides of the calcaneus.

The next step was to decide the locations of the pressure sensors. The first attempt made was to try and target 6 key nerve bundles within the foot including; plantar, deep peroneal, dorsal cutaneous, sural, fibular, and saphenous (Figure 33).

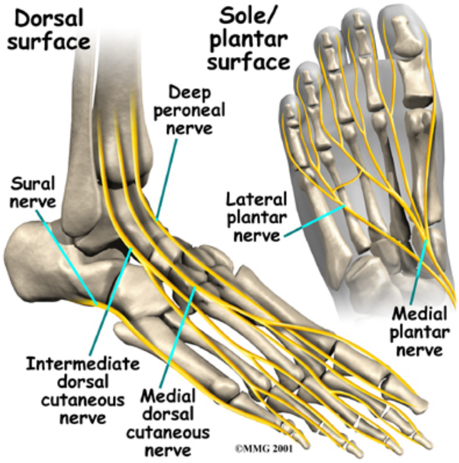


Figure 33. Key nerve bundles within the foot.

It was more favorable to target the highest pressure points within the boot, rather than the key nerve bundles. This is because the highest pressure points in the boot can lead to lack of blood flow and nerve damage. The foot was then mapped according to expected highest pressure points to find the ideal pressure sensor placement (Table 4).

Table 4. Pressure mapping results.

Sensor Location	Mean Voltage [mV]
Medial Calcaneus	4736.8
Lateral Calcaneus	4603.8
Back of Heel (Low)	0 (no contact)
Back of Heel (High)	4661.2
Lateral Pinky Toe	3926.4
Top of Foot	4113.8

Pressure mapping results indicated that the calcaneus and the top of the foot were the highest pressure points during the boot tightness. This is expected since the calcaneus pad is the primary securer of the foot when distracting the hip, and the top of the foot lies directly under the BOA clip. The calcaneus and underneath the BOA clips were determined to be the highest pressure points on the foot while the boot is on (not distracted).

Originally, the pressure sensors were planned to be sewn directly into the sock and disposed with every surgery. The idea of creating a separate sock to be wrapped around the ankle before the surgical sock was also an option. This was eliminated because it adds a new step to the surgical process. Sensors were later changed to be placed directly on the boot itself. This is to eliminate production costs of sewing in the sensors and reduce the cost of buying new sensors for each surgery.

Chapter 5. Final Design

Based off testing and the Pugh matrices in Appendix E, 4 sensors were selected and implemented to detect nerve damage and heel slip in the boot. Heel slip is detected using a laser displacement sensor. Three pressure sensors are used to properly set-up the tightness of the boot which can help limit nerve damage. (Figure 34).

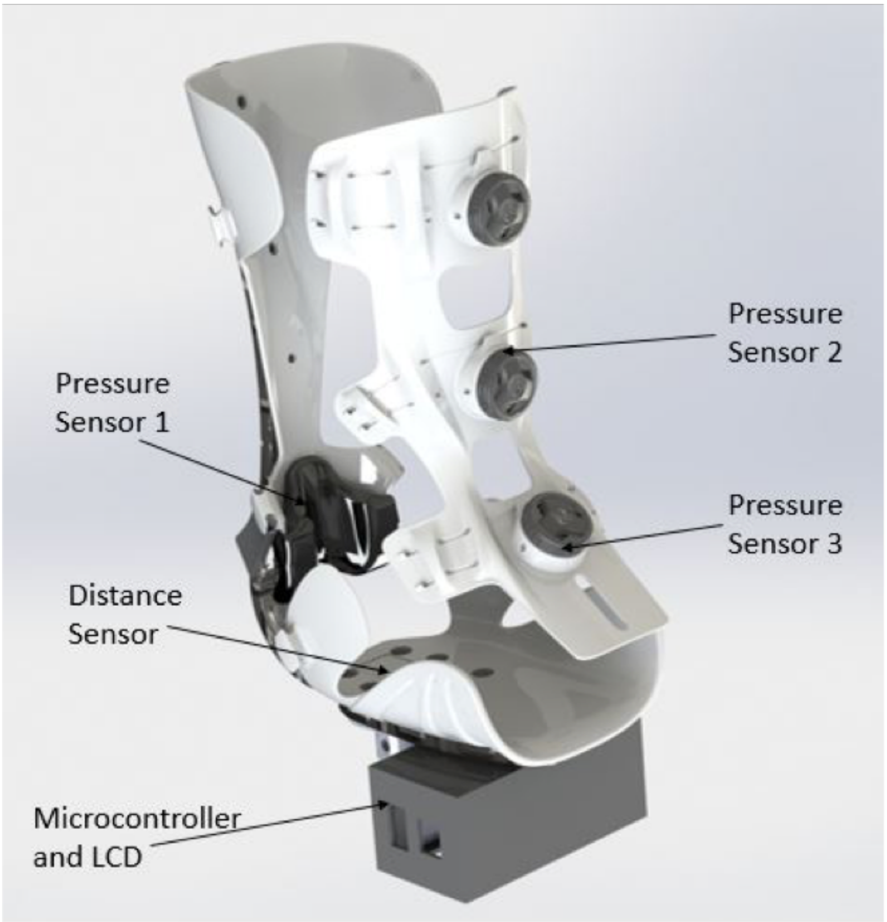


Figure 34. Overview of sensor placement

The final design was all manufactured after the sensors were selected and functioning with the Arduino separately. The sole of the boot was manufactured through a high-end 3D-printing company using custom SLS to maintain the strong, yet flexible design. The pocket was designed and manufactured to be flush with the top of the sole to make sure the user experiences no further complications. The wires for the distance sensor are fed through the tunnel which then leads up to the microcontroller. The final design also features pressure sensors attached to the boot to help offer insight when the boot is tight enough. All sensor wires lead back to the microcontroller, which is fixed on the bottom of the boot.

On the bottom of the boot is the 3D printed housing that is screwed in using the same through holes used to attach the sole to the metal spine. The box has two hinges glued on to allow the lid to rotate, with a tab sticking out being used to open and close the lid. There is a cut out the size of the 20x4 LCD screen to give the surgeon the option between the Bluetooth screen or the LCD screen. This design can be seen below in Figure 35.

Figure 35. LCD and Microcontroller Housing

While nerve damage itself is difficult to detect, the pressure sensors integrated into the boot help reduce the possibility of the boot becoming too tight which in return causes nerve damage. These sensors help to eliminate the cost associated with nerve damage and any discomfort patients may have post-surgery. The estimated cost for the distance sensor, pressure sensors and all of the hardware comes out to be $181.32 (Table 5).

Table 5. Cost Breakdown

Hardware Components	
Arduino Uno	$87.90
Adafruit Bluefruit LE SPI Friend – Bluetooth Low Energy (BLE)	$17.50
RGB backlight positive LCD 20x4	$24.95
Piezo Buzzer – PS1240	$1.50
Round Force Sensitive Resistor (FSR) – Interlink 402 (3x)	$21.00
Adafruit VL6180X Time of Flight Distance Ranging Sensor	$13.95
Parts Express 9V Battery Clip	$3.26
9V Battery	$1.30
Protoshield – Stackable Version R3	$9.95
Total Cost:	$181.31

1. Nerve Damage

Because nerve damage cannot be directly monitored or measured, three pressure sensors were incorporated in the final design to mitigate overtightening of the boot. Overtightening can cause pressure points that will result in nerve damage, so the implementation of pressure sensors can help avoid nerve damage altogether.

a. Pressure Sensor

The pressure sensor was selected because it satisfies the customer and engineering requirements. While it can be used to determine a threshold for nerve damage, it can also help surgeons develop a more structured method to tighten the boot. The pressure sensor's versatility makes it a promising option for this application.

Three pressure sensors were placed on the boot based on data collected in pressure mapping tests. Pressure mapping was performed using six pressure sensors on various parts of the foot. The final three locations were determined by locating the three points with the highest pressure values. The first 2 sensors are placed underneath the center and lower boa clips (Figure 36). The third pressure sensor is placed underneath the calcaneus pad (Figure 37).

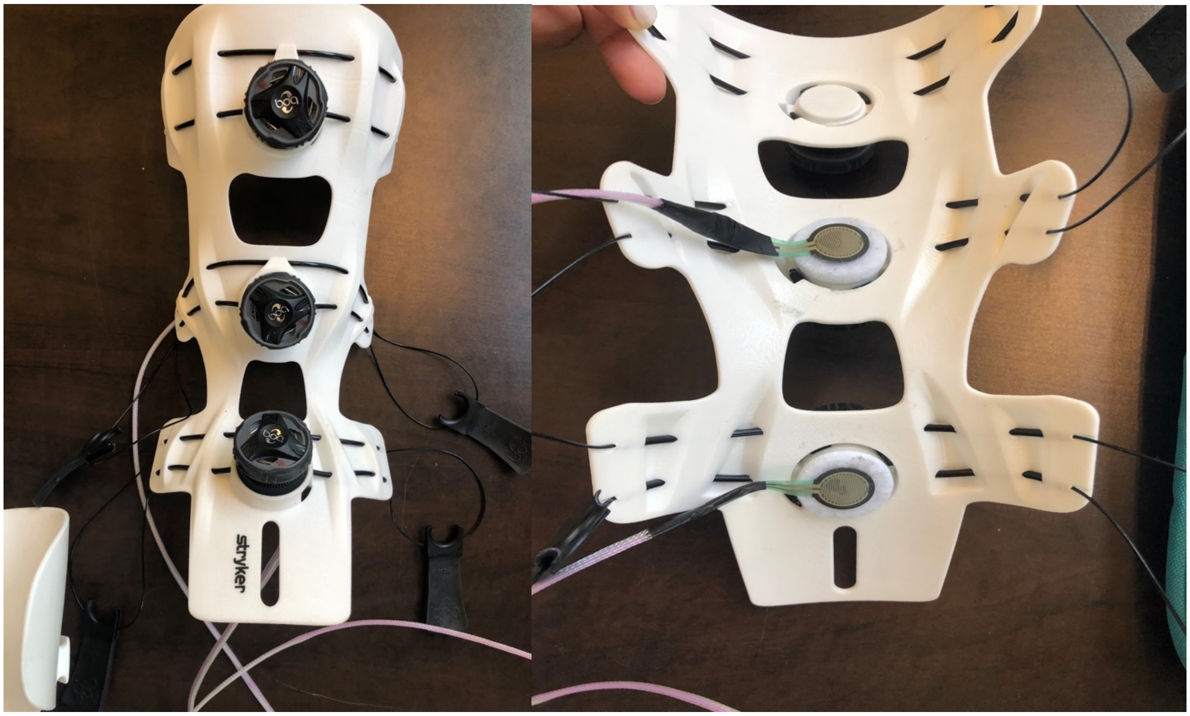


Figure 36. Sensor placement underneath BOA clips

Figure 37. Sensor placement on calcaneus pad

The combination of these 3 pressure sensors will be used during setup of the procedure to determine how tight the boot should be prior to distraction. After pressure sensor attachment, the 5 team members tested the pressure sensors in order to establish a range of acceptable pressure values prior to distraction. Values were recorded twice for each subject at 3 in-lbs of torque, as measured by a torque gage specific to the boa clips. Acceptable values were set from the mean data point to the 95th percentile (Table 6).

Table 6. Acceptable pressure sensor ranges prior to distraction.

Sensor Placement	**Minimum Pressure [N]**	**Maximum Pressure [N]**
1 - Calcaneus	55.12	82.63
2 – Shin BOA	14.37	26.50
3 – Foot BOA	38.49	54.32

As soon as the pressure is within the acceptable range, a speaker will make a beeping noise to alert the surgeon that the correct pressure range has been met.

This data was taken for only 5 subjects all within 2 years of each other. Further testing will need to be performed on a more representative demographic of hip arthroscopy patients in order to ensure this range will be successful.

During the surgery, all 3 pressure sensors will feed live data to a tablet in the room. However, the calcaneus pressure sensor will have a dual use of monitoring heel slip. During testing it was noted that as the heel slips past the calcaneus (the defined critical point) the calcaneus pressure sensor drops to zero. If this happens during the operation (post distraction) then an alarm will sound alerting the operating room staff that the calcaneus pad has lost grip of the heel, and the foot is slipping as a result.

(Manufacturing description) The two pressures sensors under the BOA clips were glued onto elevated felt pads. These felt pads are the same diameter as the boot surface under the BOA clips. The felt pads provide an elevated surface for the pressure sensors to read off of. It was found that if the pressure sensors are placed directly on the boot material, they drop to a much lower reading once 3 in-lbs of torque is reached on the BOA clips. The drop-in pressure reading was attributed to bending of the plastic boot material which decreases the load experienced by the pressure sensor. Wire wrap was secured around the two pressure sensor wires on each of the three pressure sensors. This was secured at the ends with electrical tape.

2. Heel Slip

The laser displacement sensor used to detect heel slip is called a Time of Flight Distance Sensor. As mentioned before, this sensor works by converting the time it takes to have a laser bounce off an object and come back to the receiver into a distance. This sensor is ideal because it allows the surgeon to obtain constant feedback on the distance from the datum (where the sensor is mounted) to the object (the heel). This sensor has been tested and effectively measures the distance of the object in front of the sensor. Likewise, the sensor successfully connects to the Arduino and can be read via LCD or Bluetooth.

In order to integrate the sensor into the boot, a few modifications were made to the sole. First, a pocket was created in the sole of the boot in order for the distance sensor to have a location to be mounted. The distance sensor mounted in the pocket can be seen below in Figure 38.

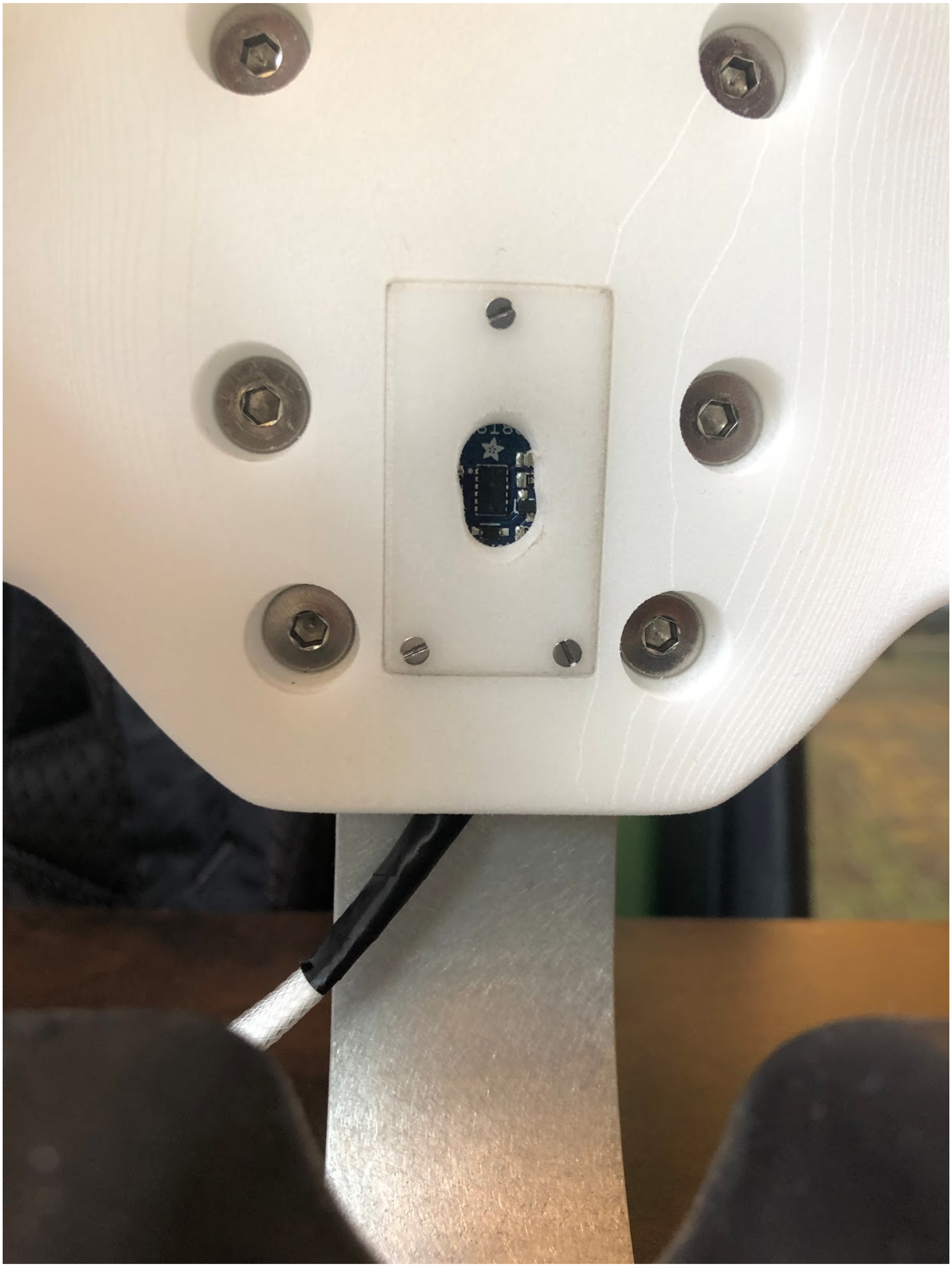


Figure 38. Placement of laser distance sensor

Having the sensor mounted in this location allows the laser to point upwards at the heel and detect how far away the heel is from the bottom of the boot. This sensor is protected by a lid which will be flush with the sole of the boot. This lid sits on three bosses which has three threaded brass inserts placed inside. The brass inserts are heat staked into position using a soldering iron. The lid also has a hole drilled which has allowed the laser to come into contact with the bottom of the heel. In order to get the wires out of the pocket, a tunnel was created to guide the wires through the sole of the boot back towards the microcontroller. An in-depth view of this integration plan can be seen below (Figure 39). Wire wrap was secured around the four distance sensor wires exiting the boot. This was secured at the ends with electrical tape. This secured the wires ran from the tunnel to the microcontroller encasing.

Figure 39. Cross-sectional view of laser distance sensor placement

3. Software and Connection

a. System Overview

The overall system architecture is shown below in Figure 40. The Arduino language, similar to the c++, is the programming language of choice for the final design. An initialization signal will start when the microcontroller is connected to power. The initialization calls for required libraries, declares variables, and creates objects. The microcontroller then runs two distinct sections of code simultaneously (distance and pressure). The tasks will contain error mitigations, such as a data buffer, in case the sensor readings yield errors.

After data collection begins, the sensor readings will be sent from the microcontroller to a Windows compatible tablet, via a Bluetooth Low Energy module, as well as be outputted numerically on a 20x4 LCD screen. The Windows tablet and LCD screen will both display numerical data and an audio/visual alarm will notify personnel if there is an anomaly. The microcontroller will be mounted on the boot itself in a custom housing. Wires will be wrapped and fed into the side of the microcontroller housing.

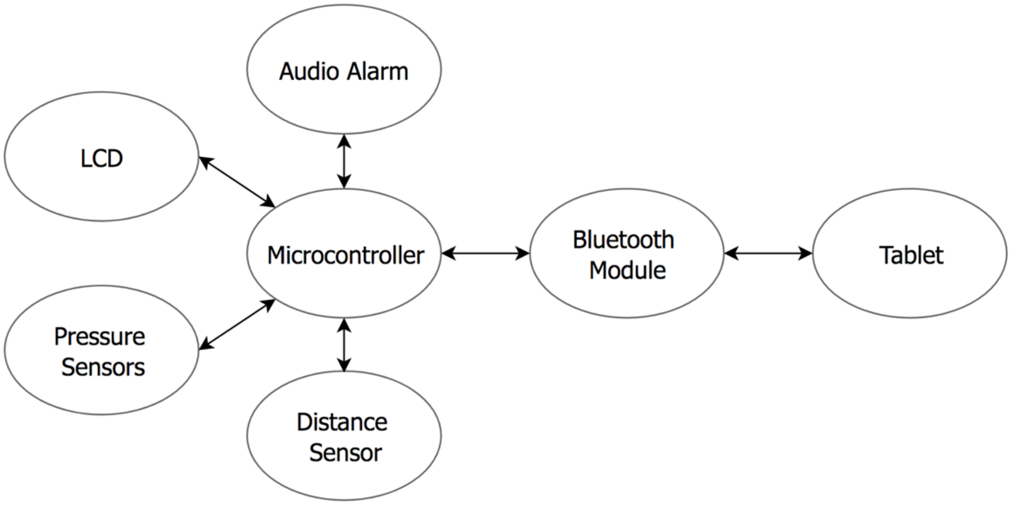


Figure 40. Overall System Architecture

b. Hardware and Connection

The overall hardware connection wiring diagram is shown below in Figure 41.

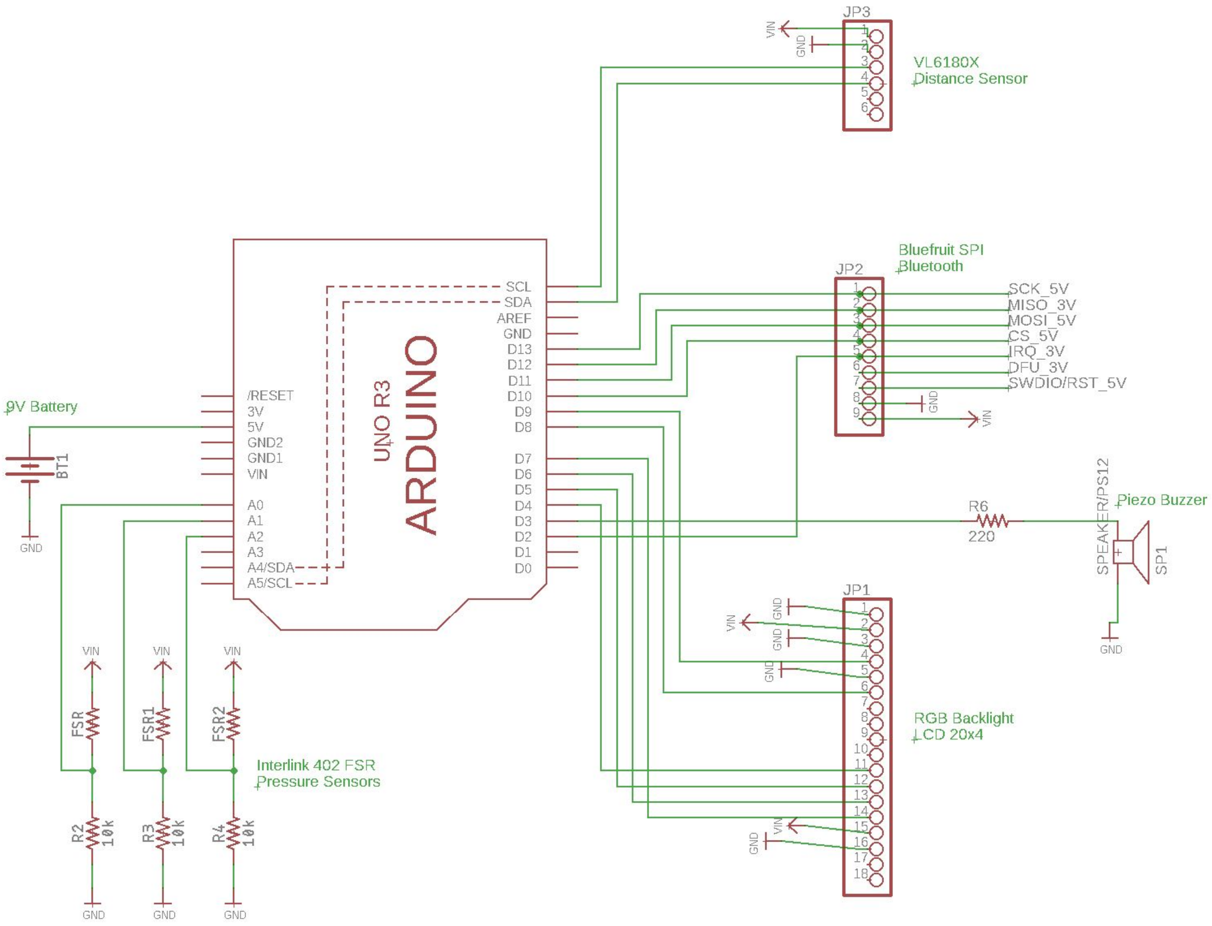


Figure 41. Overall hardware connection wiring diagram.

i. Pressure Sensors

The pressure sensors (Round Force-Sensitive Resistor (FSR) – Interlink 402) will be wired to the microcontroller as shown below in Figure 42. Each pressure sensor requires an analog input connection to the microcontroller defined in Table 7. The analog wire branches to a ground connection in series with a 10 Kohm resistor. The remaining connector tab on the sensor is to be connected to the 5V power supply on the microcontroller.

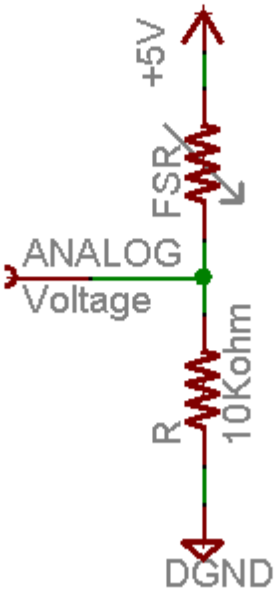


Figure 42. Schematic for pressure sensor circuit [36]

Table 7. Pressure Sensor Locations and Pinouts

Pressure Sensor Location	**Corresponding Analog Input**
Calcaneus Pad	A0
Shin BOA	A1
Foot BOA	A2

ii. Distance Sensor

The distance sensor (Adafruit VL6180X Time of Flight Distance Ranging Sensor (VL6180)) contains a chip that uses 2.8 VDC for power. The board contains a voltage regulator; therefore, it is recommended that the same power as the microcontroller is used. The sensor will be supplied with 5V power from the microcontroller as well as a ground connection to the board. The two logic pins, SCL and SDA, will be wired to the corresponding pins on the microcontroller. The pins are the clock and data pin, resp ectively.

iii. Microcontroller

Figure 43. Arduino Uno Microcontroller [37].

The microcontroller selected for the final design is the Arduino Uno, as shown in Figure 43. The microcontroller contains adequate number of pins, memory, and reliability. Detailed specifications can be found in Appendix C. The microcontroller will be powered by a 9-volt battery with a minimum life of five hours.

iv. Bluetooth Module

The Bluetooth module selected is the Adafruit Bluefruit LE SPI Friend. The module connects to the microcontroller, as shown in Figure 44. The module will be wired and located within the microcontroller housing.

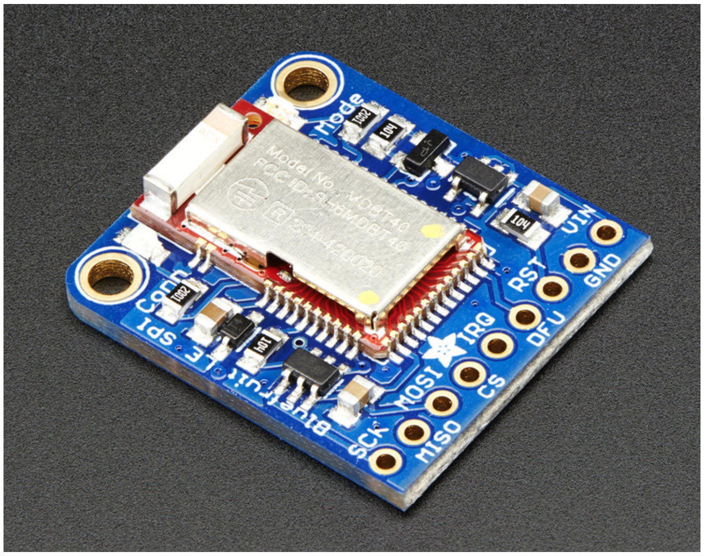


Figure 44. Adafruit Bluefruit LE SPI Friend [38].

v. Audio Alarm

The audio alarm selected is a piezo buzzer PS1240. The device makes beeps, tones, and alerts using a square wave ranging from 3 to 30 V peak-to-peak from a timer or microcontroller. Ranges for pressure and distance were established from operator testing and variability was introduced when adhesive was applied.

vi. LCD

A RGB backlight positive 20x4 LCD was selected in order to accommodate the various pressure outputs and distance. A red backlight was selected in order to enhance the color contrast and make the test more visible.

c. Software

Arduino is the programming language of choice for the final design. The language was selected because of its compatibility with the selected hardware and its ease of use. The software will follow the general process shown in Figure 45. The code can be found in Appendix H.

First, the code will be initialized when power is connected. The initialization calls the required libraries, declares variables, and creates objects. The two object declarations allow the main code to use methods from the external distance and pressure classes. After creating the objects and running the respective constructors, the code moves on to the setup function. Within the setup, the Bluetooth low energy module is initialized.

After completing the setup, the code continuously runs the loop. The loop calls methods from the distance class and the pressure class. The order of operations performed in the distance class and pressure class are similar. Each class reads the sensor data, performs the required mathematic operations on the data, and prints the data to the LCD and BLE module. The classes also trigger the piezo buzzer at certain conditions. For the distance sensor, the buzzer is triggered when the distance reading is within a certain range. For the pressure sensors, the buzzer is triggered one time per sensor when a set pressure value is reached. The data is transmitted over BLE to the Bluefruit app, designed to be compatible with the Adafruit Bluefruit LE SPI Friend.

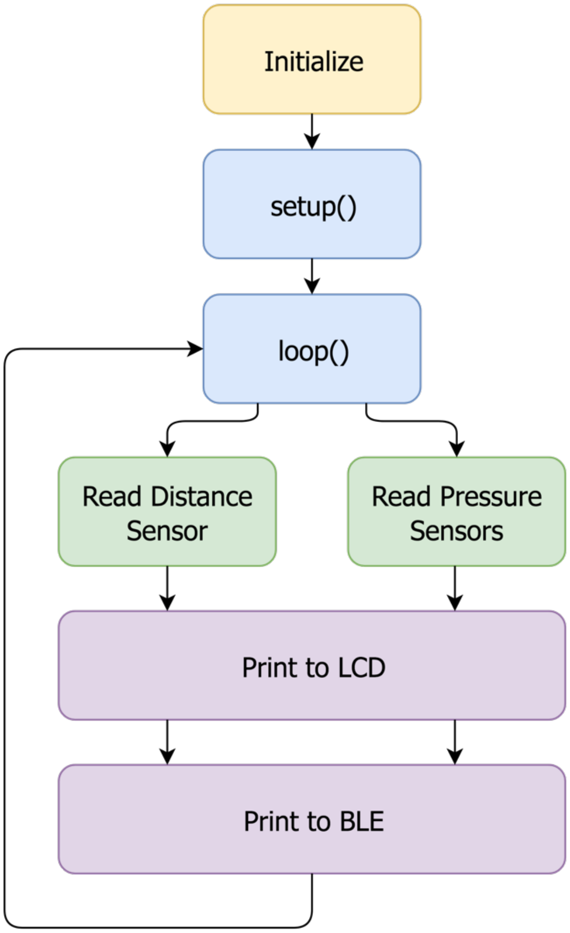


Figure 45. Software Overview

Chapter 6. Product Realization

Although most of this project revolved around research and testing, when the final assembly time came around there were a few manufacturing tasks. In order to manufacture the boot, a 3D printer was required to create all of the organic surfaces on the sole of the boot. Once the sole was printed, all of the brass inserts were heat staked down to allow the lid to be screwed down. All of the sensors were threaded through a sleeve to keep all of the wires organized. Once all of the wires were in place, the sensors were glued down to cement their position. Then came the final assembly, which involved screwing down the lid over the distance sensor and screwing the LCD/Microcontroller housing onto the bottom of the boot. Finally, a hole was drilled on the lid to allow the laser from the distance sensor to contact the heel.

A manufacturing plan for the different components for sensor integration is listed in Table 8. There are little manufacturing costs associated with the system because most of the components are off the shelf. Below is the manufacturing plan that we implemented to complete the assembly of this product.

Table 8. Manufacturing Plan

JOB	**COST**	**TIME**	**Equipment REQUIRED**
Wiring Distance Sensor	$0	6 hours	-
Wiring Pressure Sensors	$0	4 hours	Soldering Iron
Reprint Boot	$350	3 weeks	Stryker 3D Printer
Print Housing for Microcontroller	$0	2 weeks	3D Printer
Mount Housing	$0	2 hours	Allen Wrench
Software Development	$0	2 months	-

For future manufacturing, in order to make assembly easier, it is recommended to increase the tolerances on the lid for the pocket on the sole of the boot. This will allow the lid to slip out much easier, because currently it is press fitted down. It is also recommended to use a slightly bigger housing to make maintenance on the Arduino easier.

In terms of producing this product in the future, it will be more expensive due to 3D printing the boot. However, because this product is not mass produced, it does not make sense to spend money and make a mold. While the sensors and hardware used for this project were bought on a budget, for future production, more robust hardware will be used which will increase cost. This must be accounted for when considering manufacturing down the road.

Design Verification

The design verification plan and report (DVPR) is outlined in Appendix G. The plan outlines test methods and reports test details and results. The DVPR shows which sensors and devices passed the tests necessary to meet the design requirements. These tests aided the team in making conclusions about the final design configuration.

a. Pressure Sensors

Pressure sensors were tested throughout the project and following final integration into the boot. Pressure sensor testing is detailed in the preliminary testing section. An overview of the tests performed includes pressure mapping and torque to pressure correlation. Pressure mapping resulted in the decision to place 2 pressure sensors under the BOA clips, and one pressure sensor under the calcaneus since these were discovered the be the highest pressure points on the foot from the boot. A 3 in-lbs torque to pressure correlation was set based off of data from 5 group members. Pressure ranges were set from the mean to the 95th percentile pressure values for all 3 sensors. This will be utilized during the setup phase of the operation. Testing also revealed that calcaneus pressure will drop to 0 N once the heel has lost contact with the calcaneus pad indicating heel slip.

b. Heel Slip

In order to test the distance sensor, the sensor was configured to the Arduino and had measurements output to the LCD. To test the accuracy of the sensor, and verify the tolerance of ¼ inch, the distance sensor was mounted on the fixed end of a caliper. A flat surface was mounted on the slide of the caliper, with the distance sensor laser hitting the flat surface. The distance between the two parts were then stepped, and an exact measurement was compared with what was being read on the LCD. This test verified that the distance sensor being used is in fact within a ¼ inch tolerance and would be a suitable sensor for our application.

c. Software and Connection

Items number 7 and 9 from Appendix G specified the tests related to the microcontroller and software connection. First, battery life was successfully tested by running the full assembly for five hours to verify the five-hour battery life requirement. Then, the full assembly was tested with the Bluetooth connection located in various positions from the microcontroller, all within a 15-foot radius from the assembly. The test was a pass/fail test to confirm adequate connection. The software functionality was observed through the LCD screen, Bluetooth application, and/or serial monitor, depending on the stage of development.

The hardware connection verification was done throughout the prototyping through visual verification. The microcontroller, Bluetooth app, and distance sensor have built in lights to indicate that the component is receiving power.

Chapter 7. Conclusions and Recommendations

This design uses pressure sensors to address overtightening, which can lead to nerve damage. Current devices used to measure blood flow are not sensitive enough, so pressure monitoring is used as an alternative in the current design. Heel slip is monitored using a distance sensor, which is fully integrated in the boot. This design uses an Arduino Uno and transmits data to a Bluetooth app on a tablet. Future designs should include a more robust microcontroller and more accurate sensors.

The next steps for this product include further clinical testing and app development. Stryker may develop a custom app for this product if they wish to move forward with tablet integration. If a blood flow sensor is developed in the future, this could be integrated with the microcontroller to obtain more precise measurements. If a simpler design is desired, the pressure sensor located in the calcaneus pad can be used to monitor heel slip in the second generation of this product.

Acknowledgements

The team would like to thank Jeremy Graul and Adi Chola from Stryker Endoscopy for their mentorship and support throughout the course of the project. The team would also like to thank Ryan Dahlby from Stryker for sponsoring our project. The team would like to thank Shannon Conley from Koven Technology for his technical support with the vascular doppler trial. The team would like to thank River Drake from Innovation Sandbox for expediting 3D prints of the redesigned boot. The team would like to thank Charlie Refvem for assisting with the software programming. And lastly, the team would like to thank Interdisciplinary Senior Project Advisor, Lily Laiho, for the continuous support throughout the project.

Appendices

Appendix A: Quality Function Deployment
Appendix B: Pugh Matrices
Appendix C: Vendor Supplied Component Specifications and Data Sheets
Appendix D: Design Flowchart
Appendix E: Gantt Chart
Appendix F: Hazard Identification Checklist
Appendix G: Design Verification Plan and Report
Appendix H: Arduino Code
Appendix I: References

Appendix A: Quality Function Deployment

				Engineering Requirements										Benchmarks		
			Weight	Accuracy of Heel Displacement	Boot Weight	Cleaning and Disinfection	Force to Withstand	Cost	Wireless Range	Battery Life	Signal to Noise Ratio (SNR)	Microcontroller Waterproof	Windows Compatible	Arthrex Hip Distraction	Foot Plate	Active Heel Traction Boot
Customer Requirements	Doctor	Detect Heel Slip	5	9			9	1			3			3	2	3
		Ease of Use	4	3	9	9	7	1			3	4	9	4	2	3
		Simple User Interface	4	7				1	8		2		8		N/A	
		Detection of Blood Flow	5					1		2					N/A	
		Visual/Auditory Warning	3	3				1		5	4				N/A	
		Spatial Awareness	4		7			1						3	4	3
		Minimal Change in Procedure	4					1						5	5	5
		Affordable	1		2	7		1			4	3	7	1	4	3
		Sanitary	3			9		1				8		5	5	5
		Water Resistant	1					1				9		3	3	3
		Wireless	5					1	9		7				N/A	
		Lifespan	2			8	6	1		9				3	3	3
	Patient	Safety	5	9	2	8	9	1				4		5	5	5
		Comfort	3	9	6		8	1						4	1	3
		Units														
		Targets														
		Benchmark #1														
		Benchmark #2														
		Importance Scoring		166	94	126	154	49	77	43	86	72	75			
		Importance Rating (%)		100	57	76	93	30	46	26	52	43	45			
● = 9		Strong Correlation														
○ = 3		Medium Correlation														
Δ = 1		Small Correlation														
Blank		No Correlation														

Appendix B: Pugh Matrices

Nerve Damage Sensing Decision Matrix

	Concept				
Criteria	**Nerve Conduction Velocity**	**Stimulate and Measure Force**	**Stimulate and Measure EMG**	**Pressure Sensors**	**Somatosensory Evoked Potential Test (SEP)**
Function	Datum	-	+	-	S
Ease of Use		-	-	-	-
Affordable		S	S	+	S
Safety		S	S	+	S
Lifespan		-	S	-	S
Simple User Interface		-	+	S	-
Sanitary		S	S	S	S
Water Resistant		+	S	S	S
No Change in Procedure		S	S	S	-
Sum +		1	2	2	0
Sum -		4	1	3	3
Sum S		4	6	4	6

Blood Flow Sensing Decision Matrix

	Concept			
Criteria	**Pulse Oximeter**	**Laser Doppler**	**Temperature Sensing**	**Arterial Brachial Index**
Function	Datum	+	S	+
Ease of Use		-	S	-
Affordable		-	+	S
Safety		S	S	S
Lifespan		+	S	S
Simple User Interface		+	-	S
Sanitary		S	S	S
Water Resistant		S	-	+
No Change in Procedure		S	S	S
Sum +		3	1	2
Sum -		2	2	1
Sum S		4	6	6

Heel Slip Sensing Decision Matrix

	Concept				
Criteria	**Distance Sensor**	**Spring Loaded Sensor**	**Go/No Go**	**Camera to Monitor**	**Doctor Finger Check**
Function	Datum	S	-	-	S
Ease of Use		-	+	-	-
Affordable		S	+	-	+
Wireless		-	S	+	+
Safety		S	-	S	-
Lifespan		S	S	+	+
Simple User Interface		S	+	-	+
Sanitary		S	S	S	-
Water Resistant		+	S	-	+
No Change in Procedure		-	S	S	-
Sum +		1	3	2	5
Sum -		3	2	5	4
Sum S		6	5	3	1

Appendix C: Vendor Supplied Component Specifications and Data Sheets

Component Summaries

(Comprehensive data sheets for components are located on source website)

Round Force-Sensitive Resistor (FSR) – Interlink 402

Source: https://www.adafruit.com/product/166

Dimensions:

- Length: 56.77mm/2.35in
- Width: 18.48mm/0.73in
- Thickness: 0.55mm/0.02in
- Weight: 0.26g/0.01oz

Size: 1/2" (12.5mm) diameter active area by 0.02" thick (Interlink does have some that are as large as 1.5"x1.5")

Resistance range: Infinite/open circuit (no pressure), 100KΩ (light pressure) to 200Ω (max. pressure)

Force range: 0 to 20 lb. (0 to 100 Newtons) applied evenly over the 0.125 sq in surface area

Power supply: Any! Uses less than 1mA of current (depends on any pullup/down resistors used and supply voltage)

Adafruit VL6180X Time of Flight Distance Ranging Sensor (VL6180)

Source: https://learn.adafruit.com/adafruit-vl6180x-time-of-flight-micro-lidar-distance-sensor-breakout/overview

Product Dimensions: 20.5mm x 18.0mm x 3.0mm / 0.8" x 0.7" x 0.1"

Product Weight: 1.4g / 0.0oz

Power Pins:

- **Vin** - this is the power pin. Since the chip uses 2.8 VDC, we have included a voltage regulator on board that will take 3-5VDC and safely convert it down. To power the board, give it the same power as the logic level of your microcontroller - e.g. for a 5V micro like Arduino, use 5V
- **2v8** - this is the 2.8V output from the voltage regulator, you can grab up to 100mA from this if you like
- **GND** - common ground for power and logic

I2C Logic pins:

- **SCL** - I2C clock pin, connect to your microcontrollers I2C clock line.
- **SDA** - I2C data pin, connect to your microcontrollers I2C data line.
- Connect **Vin** to the power supply, 3-5V is fine. Use the same voltage that the microcontroller logic is based off of. For most Arduinos, that is 5V
- Connect **GND** to common power/data ground
- Connect the **SCL** pin to the I2C clock **SCL** pin on your Arduino. On an UNO & '328 based Arduino, this is also known as **A5**, on a Mega it is also known as **digital 21** and on a Leonardo/Micro, **digital 3**
- Connect the **SDA** pin to the I2C data **SDA** pin on your Arduino. On an UNO & '328 based Arduino, this is also known as **A4**, on a Mega it is also known as **digital 20** and on a Leonardo/Micro, **digital 2**

Software: Adafruit_VL6180X Library

Adafruit Bluefruit LE SPI Friend

Source: https://learn.adafruit.com/introducing-the-adafruit-bluefruit-spi-breakout/introduction

Technical Specifications:

- ARM Cortex M0 core running at 16MHz (nRF51822)
- 256KB flash memory
- 32KB SRAM
- Transport: SPI at 4MHz with HW IRQ (5 pins required)
- 5V-safe inputs (Arduino Uno friendly, etc.)
- On-board 3.3V voltage regulation
- Bootloader with support for safe OTA firmware updates
- Easy AT command set to get up and running quickly

Power Pins:

- **VIN**: This is the power supply for the module, supply with 3.3-16V power supply input. This will be regulated down to 3.3V to run the chip
- **GND**: The common/GND pin for power and logic

SPI Pins:

- **SCK**: This is the serial clock pin, connected to SCK on your Arduino or MCU
- **MISO**: This is the Master In Slave Out SPI pin (nRF51 -> Arduino communication)
- **MOSI**: This is the Master Out Slave In SPI pin (Arduino -> nRF51 communication)
- **CS**: This is the Chip Select SPI pin, which is used to indicate that the SPI device is currently in use.
- **IRQ**: This is the nRF51 -> Arduino 'interrupt' pin that lets the Arduino or MCU know when data is available on the nRF51, indicating that a new SPI transaction should be initiated by the Arduino/MCU.

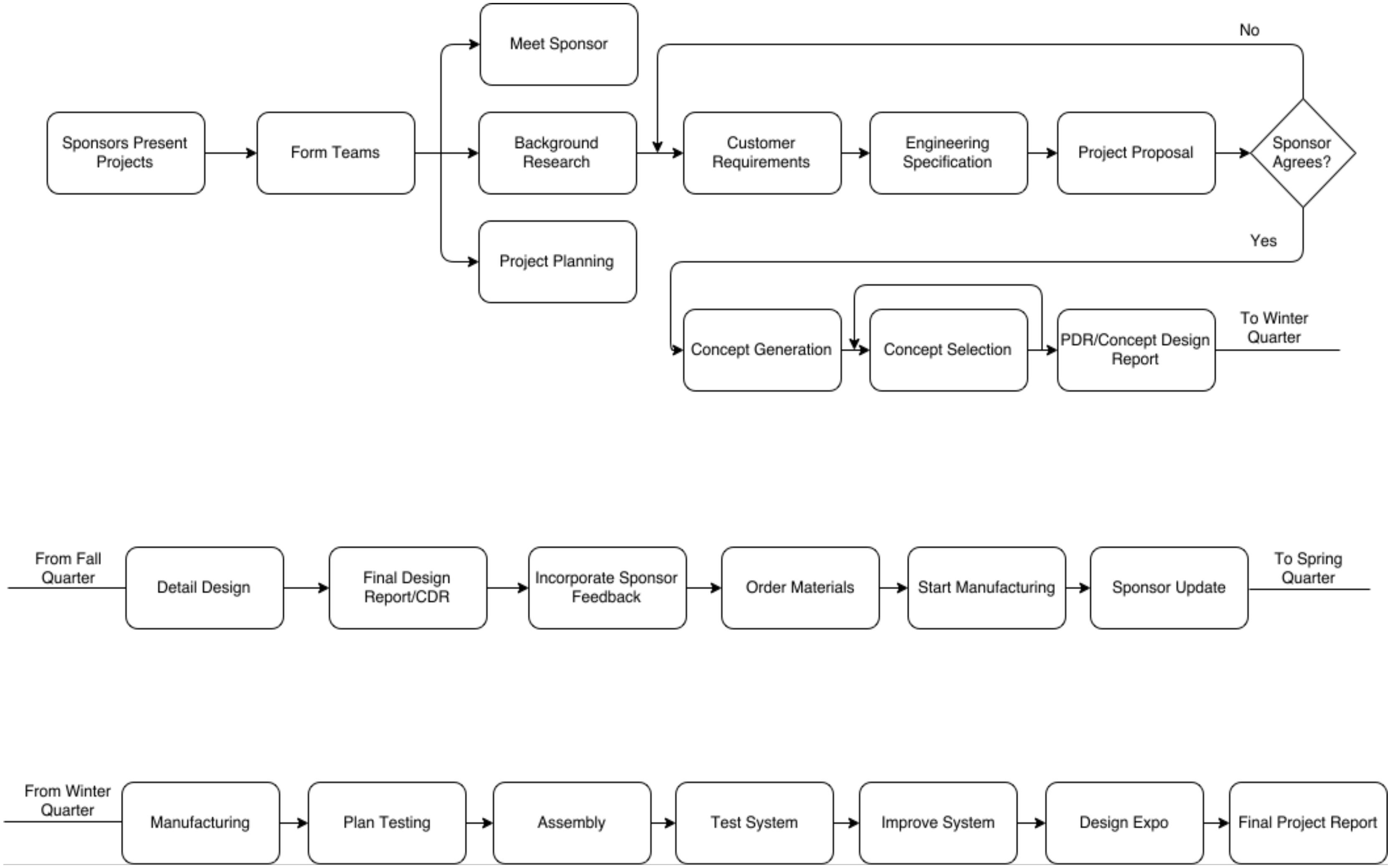
Software: Adafruit_BluefruitLE_nRF51 Library

Arduino Uno Rev3

Source: https://store.arduino.cc/arduino-uno-rev3

Microcontroller	[ATmega328P](#)
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g

Appendix D: Design Flowchart*

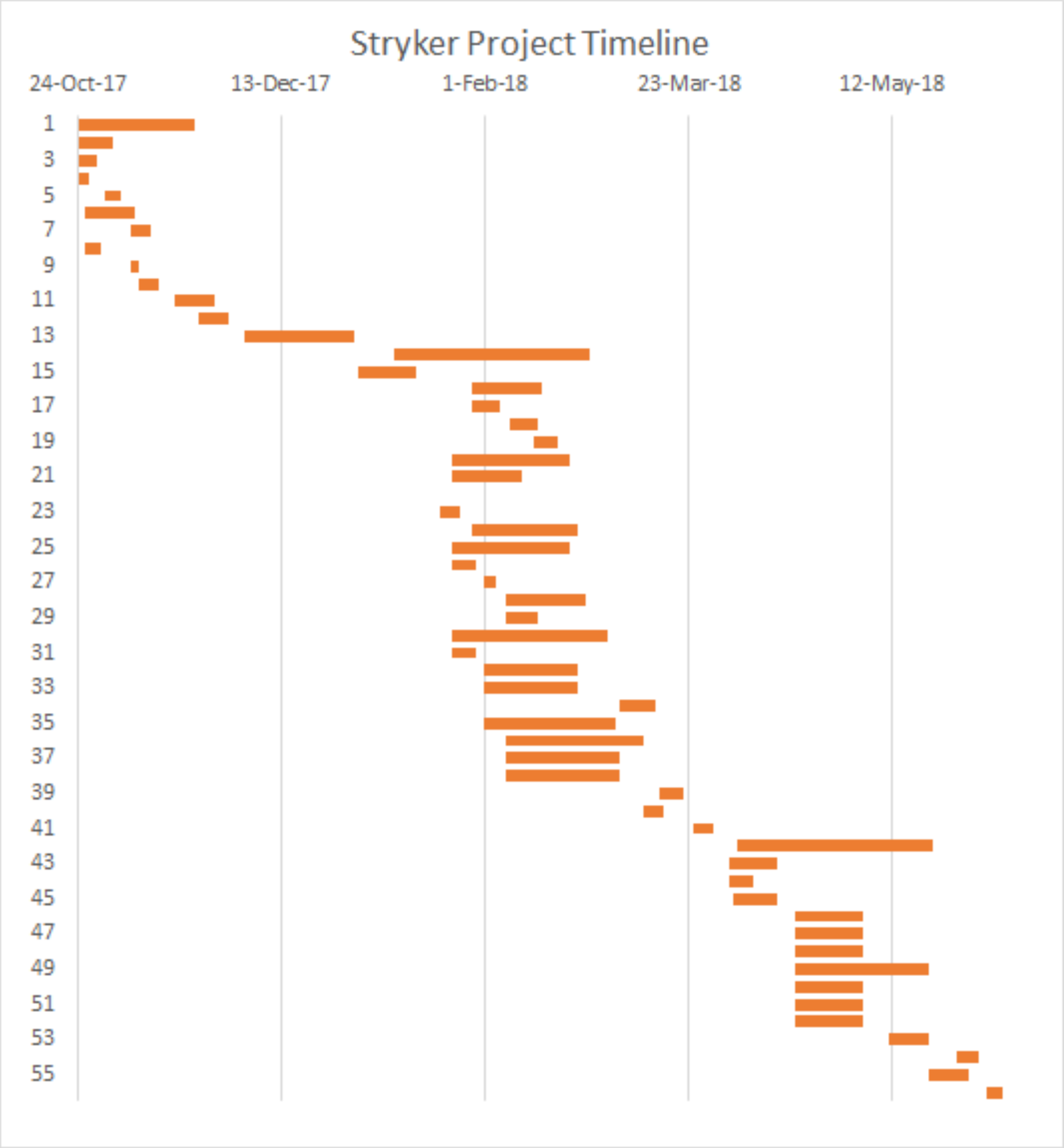


*From Senior Project lecture

Appendix E: Gantt Chart

No.	Task	Start Date	End Date	Duration
1	**Conceptual Design**	24-Oct-17	1-Dec-17	29
2	Sensor Selection	24-Oct-17	3-Nov-17	9
3	- Order Parts	24-Oct-17	30-Oct-17	5
4	- Deliverable: Conceptual Models	24-Oct-17	26-Oct-17	3
5	- Conceptual Models (with sensors arrived)	31-Oct-17	3-Nov-17	4
6	Sensor Placement	26-Oct-17	10-Nov-17	12
7	- Conceptual Models	6-Nov-17	10-Nov-17	5
8	- Deliverable: Project Schedule	26-Oct-17	31-Oct-17	4
9	- Deliverable: Decision Matrix	6-Nov-17	7-Nov-17	2
10	Deliverable: Conceptual Design Slides	8-Nov-17	14-Nov-17	5
11	Deliverable: 1st Presentation to Sponsor	17-Nov-17	30-Nov-17	10
12	Deliverable: Conceptual Design Report	23-Nov-17	1-Dec-17	7
13	Winter Break	4-Dec-17	9-Jan-18	27
14	**Critical Design**	10-Jan-18	16-Mar-18	48
15	Deliverable: Reflection Memo on Team Dynamics	1-Jan-18	18-Jan-18	14
16	Critical Design Presentation	29-Jan-18	20-Feb-18	17
17	- Deliverable: Critical Design Presentation	29-Jan-18	6-Feb-18	7
18	- Deliverable: 2nd Presentation to Sponsor	7-Feb-18	15-Feb-18	7
19	- Deliverable: Critical Design Report	13-Feb-18	20-Feb-18	6
20	Blood Flow Sensors	24-Jan-18	5-Mar-18	29
21	- Order Parts	24-Jan-18	15-Feb-18	17
22	- ABI Testing	21-Jan-18	21-Jan-18	0
23	- SpO2 % Testing	21-Jan-18	28-Jan-18	5
24	- Perfusion Index Testing	29-Jan-18	5-Mar-18	26
25	Heel Slip Sensor	24-Jan-18	5-Mar-18	29
26	- Order Parts	24-Jan-18	31-Jan-18	6
27	- Wiring Configuration	1-Feb-18	5-Feb-18	3
28	- Distance Testing	6-Feb-18	5-Mar-18	20
29	- Output Graphs	6-Feb-18	15-Feb-18	8
30	Pressure Sensors	24-Jan-18	16-Mar-18	38
31	- Order Parts	24-Jan-18	31-Jan-18	6
32	- Configuration of Multiple Sensors	1-Feb-18	5-Mar-18	23
33	- Pressure Mapping	1-Feb-18	5-Mar-18	23
34	- Pressure Integration (Sew onto Soft Pad)	6-Mar-18	16-Mar-18	9
35	- Output Data Values for all Sensors	1-Feb-18	16-Mar-18	32
36	3D Print Housing	6-Feb-18	25-Mar-18	34
37	- Microcontroller Housing Design	6-Feb-18	15-Mar-18	28
38	- Boot with Sensor Integration Design	6-Feb-18	15-Mar-18	28
39	- 3D Print	16-Mar-18	25-Mar-18	6
40	Deliverable: Project Update Memo to Sponsor	12-Mar-18	16-Mar-18	5
41	Spring Break	24-Mar-18	1-Apr-18	5
42	**Final Product**	4-Apr-18	8-Jun-18	48
43	Deliverable: Hardware and Software Demonstration	2-Apr-18	17-Apr-18	12
44	- Order Parts	2-Apr-18	9-Apr-18	6
45	- Bluetooth Integration	3-Apr-18	17-Apr-18	11
46	Finalize User Interface	18-Apr-18	10-May-18	17
47	- Software	18-Apr-18	10-May-18	17
48	- Hardware	18-Apr-18	10-May-18	17
49	Final Product Quantitative Analysis	18-Apr-18	1-Jun-18	33
50	- Heel Slip Testing	18-Apr-18	10-May-18	17
51	- Pressure Testing	18-Apr-18	10-May-18	17
52	- Blood Flow Testing	18-Apr-18	10-May-18	17
53	- Statistical Analysis	11-May-18	24-May-18	10
54	- Report Conclusions	28-May-18	1-Jun-18	5
55	Deliverable: Senior Project Expo	21-May-18	1-Jun-18	10
56	Deliverable: 3rd Presentation to Sponsor	4-Jun-18	8-Jun-18	5

*Updated 19-Feb-18


Stryker Project Timeline
24-Oct-17
13-Dec-17
1-Feb-18
23-Mar-18
12-May-18
1
3
5
7
9
11
13
15
17
19
21
23
25
27
29
31
33
35
37
39
41
43
45
47
49
51
53
55

Appendix F: Hazard Identification Checklist

SENIOR PROJECT CRITICAL DESIGN REVIEW HAZARD IDENTIFICATION CHECKLIST

Y	N	
☑	☐	Do any parts of the design create hazardous revolving, reciprocating, running, shearing, punching, pressing, squeezing, drawing, cutting, rolling, mixing or similar action, including pinch points and sheer points adequately guarded?
☐	☑	Does any part of the design undergo high accelerations/decelerations that are exposed to the user?
☐	☐	Does the system have any large moving masses or large forces that can contact the user?
☐	☑	Does the system produce a projectile?
☐	☑	Can the system to fall under gravity creating injury?
☐	☑	Is the user exposed to overhanging weights as part of the design?
☐	☑	Does the system have any sharp edges exposed?
☐	☑	Are there any ungrounded electrical systems in the design?
☐	☑	Are there any large capacity batteries or electrical voltage in the system above 40 V either AC or DC?
☑	☐	Is there be any stored energy in the system such as batteries, flywheels, hanging weights or pressurized fluids when the system is either on or off?
☐	☑	Is there any explosive or flammable liquids, gases, dust, or fuel part of the system?
☐	☑	Is the user of the design required to exert any abnormal effort and/or assume an abnormal physical posture during the use of the design?
☐	☑	Is there be any materials known to be hazardous to humans involved in either the design or the manufacturing of the design?
☐	☑	Will the system generate high levels of noise?
☐	☑	Will the product be subjected to extreme environmental conditions such as fog, humidity, cold, high temperatures, etc. that could create an unsafe condition?
☐	☑	Is it easy to use the system unsafely?
☐	☑	Is there be any other potential hazards not listed above? If yes, please explain on the back of this checklist?

Appendix G: Design Verification Plan and Report

DVP&R

Report Date: 6/7/2018
Sponsor: Stryker
Reporting Engineer: PolyHIPS

TEST PLAN										TEST REPORT			
Item No	Specification or Clause Reference	Test Description	Acceptance Criteria	Test Responsibility	Test Stage	SAMPLES TESTED		TIMING		TEST RESULTS			NOTES
						Quantity	Type	Start date	Finish date	Test Result	Quantity Pass	Quantity Fail	
1	SpO2 % Signal Strength	Run the sensor as the boot is being continually tightened and determine if the signal strength is strong and if the SpO2 % decreases in relation to time in boot	Signal Strength: Strong	Chris, Katie	Bonderson	1	Boot and Pulse Oximeter	1/21/2018	1/21/2018	Fail		1	
2	Perfusion Index Strength (30 min)	Record perfusion index while boot tightening to evaluate the strength of the signal	Signal Strength: Strong	Esther, Claire	Bonderson	1	Boot and Pulse Oximeter	2/15/2018	2/15/2018	Pass	1		
3	Perfusion Index Accuracy (2 hours)	Record perfusion index while boot tightened; there should be a decrease in relation to time in boot	0.2-20%	Esther, Claire	Bonderson	1	Boot and Pulse Oximeter	2/27/2018	3/13/2018	Fail		1	
4	Pressure Mapping	Place boot on subjects and point out largest pressure, remove boot and place pressure sensor on most compressed areas, tighten boot to surgical requirement, take pressure readings to diagnose highest pressure points within the boot.	N/A (Placement Diagnostic)	Katie, Esther	Bonderson	1	Pressure Sensors	3/8/2018	3/8/2018	Pass	1		
5	Vascular Doppler Strength	Record peak-to-peak measurements of blood flow while boot tightening to evaluate the strength of the signal	Signal Strength: Strong	Esther, Claire	Bonderson	1	Boot and Vascular Doppler	4/10/2018	4/17/2018	Pass	1		
6	Vascular Doppler Accuracy	Record perfusion index while boot tightened; there should be a decrease in relation to time in boot	0.1-16 PP	Esther, Claire	Bonderson	1	Boot and Vascular Doppler	4/17/2018	5/8/2018	Pass	1		
7	Accuracy of Heel Displacement	Create fixed distance to move from 0 to 80mm in increments of 8mm, measure distance using sensor code and verify that it is within acceptance criteria	±1/4"	Chris	Bonderson	1	Boot and Heel Sensor	4/19/2018	4/19/2018	Pass	1		
8	Torque to Pressure Correlation	Utilize Stryker's torque quantifying device to see how levels of torque on the boot's three tighteners correspond with pressure readings.	N/A (Correlate Torque to Pressure)	Katie, Esther	Bonderson	1	Pressure Sensors	5/10/2018	5/17/2018	Pass	1		
9	Wireless Range	Test the ability of the Bluetooth module to transmit data at a range of 15 feet.	15 feet	Bri	Bonderson	1	Microcontroller assembly	5/25/2018	5/25/2018	Pass	1		
10	Battery Life	Run all sensors, microcontroller, and iPad and run for 5 hours to verify battery life.	5 Hours	All	Bonderson	1	Full assembly	6/5/2018	6/5/2018	Pass	1		
11	Windows Compatible	Sensors work with Windows	Pass/Fail	Bri	Bonderson	1	Full assembly	6/5/2018	6/5/2018	Pass	1		
12	Boot Weight	Measure boot with all sensors and microcontroller in position on accurate scale	5lbs	Katie	ME Vibes Lab	1	Full assembly	6/5/2018	6/5/2018	Pass	1		

Appendix H: Arduino Code

heelslip_final.ino

```
#include "distance.h"
#include "pressure.h"
#include <LiquidCrystal.h>

#include <Arduino.h>
#include <SPI.h>
#include "Adafruit_BLE.h"
#include "Adafruit_BluefruitLE_SPI.h"
#include "BluefruitConfig.h"

#if SOFTWARE_SERIAL_AVAILABLE
  #include <SoftwareSerial.h>
#endif

#define FACTORYRESET_ENABLE         1         // from BLE example (blueart_datamode)
#define    MINIMUM_FIRMWARE_VERSION              "0.6.6"           //    from    BLE    example
(blueart_datamode)
#define MODE_LED_BEHAVIOUR          "MODE"    // from BLE example (blueart_datamode)

const int rs = 9, en = 8, d4 = 4, d5 = 5, d6 = 6, d7 = 7;
LiquidCrystal lcd = LiquidCrystal(rs, en, d4, d5, d6, d7);

//
//Adafruit_BluefruitLE_SPI ble(BLUEFRUIT_SPI_SCK, BLUEFRUIT_SPI_MISO,
//                             BLUEFRUIT_SPI_MOSI, BLUEFRUIT_SPI_CS,
//                             BLUEFRUIT_SPI_IRQ, BLUEFRUIT_SPI_RST);

Adafruit_BluefruitLE_SPI ble(
                        BLUEFRUIT_SPI_CS,
                        BLUEFRUIT_SPI_IRQ,
                        BLUEFRUIT_SPI_RST);  // bluefruit object, uses BluefruitConfig.h here


//Creating the distance object for DistanceSensor class
DistanceSensor distance = DistanceSensor(&lcd, &ble);
```

```
 //Creating the pressure object for PressureSensor class
PressureSensor pressure = PressureSensor(&lcd, &ble);

// A small helper
void error(const __FlashStringHelper*err) {
  //Serial.println(err);
  while (1);
}


void setup() {
  //Serial.begin(115200);
  lcd.setCursor(0,3);
  lcd.print("SETUP");
  /* ...software SPI, using SCK/MOSI/MISO user-defined SPI pins and then user selected
CS/IRQ/RST */
  while (!Serial);                        // required for Flora & Micro, from BLE example
(blueart_datamode)
  delay(500);
  Serial.begin(115200);

  if ( !ble.begin(VERBOSE_MODE) )
  {
    error(F("Couldn't find Bluefruit, make sure it's in CoMmanD mode & check wiring?"));
  }
  Serial.println( F("OK!") );

  if ( FACTORYRESET_ENABLE )
  {
    /* Perform a factory reset to make sure everything is in a known state */
    Serial.println(F("Performing a factory reset: "));
    if ( ! ble.factoryReset() ){
      error(F("Couldn't factory reset"));
    }
  }

  // Set module to DATA mode
  Serial.println( F("Switching to DATA mode!") );
  ble.setMode(BLUEFRUIT_MODE_DATA);
}
```

```
void loop() {

  distance.getDistance();
  distance.printDistanceData();
  pressure.getPressure();
  pressure.printPressureData();
  delay(500);
}
```

distance.h

```
#ifndef _DISTANCE
#define _DISTANCE   // adding include guards

#include <LiquidCrystal.h>
#include <Wire.h>
#include "Adafruit_VL6180X.h"

#include <Arduino.h>
#include <SPI.h>
#include "Adafruit_BLE.h"
#include "Adafruit_BluefruitLE_SPI.h"
#include "bluefruit.h"

#if SOFTWARE_SERIAL_AVAILABLE
  #include <SoftwareSerial.h>
#endif


#define NUM_READINGS 5

class DistanceSensor
{
  public:
    DistanceSensor(LiquidCrystal* ptr_lcd, Adafruit_BluefruitLE_SPI* ptr_ble);
    void getDistance();
    void printDistanceData();

  private:
    void distanceThreshold();
```

```
    void readDistanceSensor();
    void errorManager();
    void distanceMath();
    void storeData();
    void applyThreshold();

    Adafruit_VL6180X vl;
    LiquidCrystal* p_lcd;
    Adafruit_BluefruitLE_SPI* p_ble;
    int m_range = 0;
    int m_state = 0;
    int m_readings[NUM_READINGS] = {0};   // the readings from the analog input
    int m_readIndex = 0;                  // the index of the current reading
    int m_total = 0;                      // the running total
    int m_average = 0;                    // the average
    int m_average_old = 0;                // the old average
    int m_change = 0;                     // delta of distance

};

#endif
```

distance.cpp

```
#include "distance.h"
int piezoPin = 3;

// Constructor for the distance sensor
DistanceSensor::DistanceSensor(LiquidCrystal* ptr_lcd, Adafruit_BluefruitLE_SPI* ptr_ble)
{
  //Create an object of Adfruit_VL6180X class
  vl = Adafruit_VL6180X();
  p_ble = ptr_ble;
  p_lcd = ptr_lcd;
  p_lcd->begin(20,4);
  //Serial.begin(115200);
  //Serial.print("Adafruit VL6180x test!");
}


void DistanceSensor::getDistance()
```

```
{
 vl.begin();
 //readDistanceSensor();
 distanceMath();
}


void DistanceSensor::readDistanceSensor()
{
 m_range = vl.readRange();
 m_state = vl.readRangeStatus();
}

void DistanceSensor::distanceMath()
{
 m_range = vl.readRange();
 m_state = vl.readRangeStatus();

 if (m_state == VL6180X_ERROR_NONE) {
  // subtract the last reading:
  m_total = m_total - m_readings[m_readIndex];
  // read from the sensor:
  m_readings[m_readIndex] = vl.readRange();
  // add the reading to the total:
  m_total = m_total + m_readings[m_readIndex];
  // advance to the next position in the array:
  m_readIndex = m_readIndex + 1;

  // if we're at the end of the array...
  if (m_readIndex >= NUM_READINGS) {
   // ...wrap around to the beginning:
    m_readIndex = 0;
   }

  // calculate the average:
  m_average = m_total / NUM_READINGS;
  m_change = m_average - m_average_old;
  m_average_old = m_average;
  if (m_average >= 45 && m_average <= 65) {
   tone(piezoPin, 600, 500);
```

```
    }
   }

  else
  {
   m_average=0;
  }
}

void DistanceSensor::printDistanceData()
{
  Serial.println("Distance");
  Serial.println(m_average);
  p_lcd->setCursor(0,0);
  p_lcd->print("Dist:             ");
  p_lcd->setCursor(7,0);
  p_lcd->print(m_average);
  p_ble->println("DISTANCE:");
  p_ble->print(m_average);
  p_ble->println(" [mm]");
  p_ble->println("      ");
}
```

pressure.h

```
#ifndef _PRESSURE
#define _PRESSURE   // adding incude guards

#include <LiquidCrystal.h>
#include <Wire.h>
#include "Arduino.h"

#include <Arduino.h>
#include <SPI.h>
#include "Adafruit_BLE.h"
#include "Adafruit_BluefruitLE_SPI.h"
#include "bluefruit.h"

#if SOFTWARE_SERIAL_AVAILABLE
  #include <SoftwareSerial.h>
```

```
#endif


class PressureSensor
{
  public:
    PressureSensor(LiquidCrystal* ptr_lcd, Adafruit_BluefruitLE_SPI* ptr_ble);
    void getPressure();
    void printPressureData();
    void readAnalog();
    void voltageMap();
    void performMath();

  private:

    int m_fsrPins[3] = {0,1,2};
    int m_fsrReading[3];              // the analog reading from the FSR resistor divider
    int m_fsrVoltage[3];             // the analog reading converted to voltage
    unsigned long m_fsrResistance[3];  // The voltage converted to resistance, can be very big
so make "long"
    unsigned long m_fsrConductance[3];
    float m_fsrForce[3];             // Finally, the resistance converted to force

    LiquidCrystal* p_lcd;
    Adafruit_BluefruitLE_SPI* p_ble;

};

#endif
```

pressure.cpp

```
#include "pressure.h"
int flag0;
int flag1;
int flag2;

int piezoPin2 = 3;

PressureSensor::PressureSensor(LiquidCrystal* ptr_lcd, Adafruit_BluefruitLE_SPI* ptr_ble)
```

```
{
  p_ble = ptr_ble;
  p_lcd = ptr_lcd;
  p_lcd->setCursor(0,1);
}

void PressureSensor::getPressure()
{
  readAnalog();
  voltageMap();
  performMath();
}

void PressureSensor::readAnalog()
{
  for (int i = 0; i < 3; i = i + 1)
  {
    m_fsrReading[i] = analogRead(m_fsrPins[i]);
  }
}


void PressureSensor::voltageMap()
{
  for (int i = 0; i < 3; i = i + 1)
  {
    m_fsrVoltage[i] = map(m_fsrReading[i], 0, 1023, 0, 5000);
  }

}

void PressureSensor::performMath()
{
  for (int i = 0; i < 3; i= i + 1)
  {
    m_fsrResistance[i] = 5000 - m_fsrVoltage[i];    // fsrVoltage is in millivolts so 5V = 5000mV
    m_fsrResistance[i] *= 10000;                    // 10K resistor
    m_fsrResistance[i] /= m_fsrVoltage[i];
    m_fsrConductance[i] = 1000000;                  // we measure in micromhos so
    m_fsrConductance[i] /= m_fsrResistance[i];
```

```
   if (m_fsrConductance[i] <= 1000)
   {
     m_fsrForce[i] = m_fsrConductance[i] / 80.00;
     if (m_fsrForce[0] > 3 && flag0 == 0) {
       tone(piezoPin2, 1000, 500);
       flag0 = 1;
     }
     if (m_fsrForce[1] > 3 && flag1 == 0) {
       tone(piezoPin2, 500, 500);
       flag1 = 1;
     }
     if (m_fsrForce[2] > 3 && flag2 == 0) {
       tone(piezoPin2, 200, 500);
       flag2 = 1;
     }
   }
   else
   {
     m_fsrForce[i] = m_fsrConductance[i] - 1000;
     m_fsrForce[i] /= 30.00;
     if (m_fsrForce[0] > 3 && flag0 == 0) {
       tone(piezoPin2, 1000, 500);
       flag0 = 1;
     }
     if (m_fsrForce[1] > 3 && flag1 == 0) {
       tone(piezoPin2, 500, 500);
       flag1 = 1;
     }
     if (m_fsrForce[2] > 3 && flag2 == 0) {
       tone(piezoPin2, 200, 500);
       flag2 = 1;
     }
   }
 }
}


void PressureSensor::printPressureData()
{
```

```
 //print to LCD
 p_lcd->setCursor(0,2);
 p_lcd->print("Pressure:");
 p_lcd->print(" P1:     ");
 p_lcd->setCursor(13,2);
 p_lcd->print(m_fsrForce[0]);
 p_lcd->setCursor(0,3);
 p_lcd->print("P2: ");
 p_lcd->setCursor(3,3);
 p_lcd->print(m_fsrForce[1]);
 p_lcd->setCursor(10,3);
 p_lcd->print("P3:");
 p_lcd->print(m_fsrForce[2]);
//  //print to BLE
//  p_ble->println("PRESSURE_SENSORS:  ");
//  p_ble->print( m_fsrForce[0]);
//  p_ble->println(" [N]");
//  p_ble->print(m_fsrForce[1]);
//  p_ble->println(" [N]");
//  p_ble->print( m_fsrForce[2]);
//  p_ble->println(" [N]");
//  p_ble->println("---------------------");

}
```

BluefruitConfig.h

```
// COMMON SETTINGS
// ----------------------------------------------------------------------------------------------
// These settings are used in both SW UART, HW UART and SPI mode
// ----------------------------------------------------------------------------------------------
#define BUFSIZE                        128   // Size of the read buffer for incoming data
#define VERBOSE_MODE                   true  // If set to 'true' enables debug output
#define BLE_READPACKET_TIMEOUT         500   // Timeout in ms waiting to read a response


// SOFTWARE UART SETTINGS
// ----------------------------------------------------------------------------------------------
// The following macros declare the pins that will be used for 'SW' serial.
// You should use this option if you are connecting the UART Friend to an UNO
```

```
// ----------------------------------------------------------------------------------------------
//#define BLUEFRUIT_SWUART_RXD_PIN       9    // Required for software serial!
//#define BLUEFRUIT_SWUART_TXD_PIN       10   // Required for software serial!
//#define BLUEFRUIT_UART_CTS_PIN         11   // Required for software serial!
//#define BLUEFRUIT_UART_RTS_PIN         -1   // Optional, set to -1 if unused


// HARDWARE UART SETTINGS
// ----------------------------------------------------------------------------------------------
// The following macros declare the HW serial port you are using. Uncomment
// this line if you are connecting the BLE to Leonardo/Micro or Flora
// ----------------------------------------------------------------------------------------------
#ifdef Serial1    // this makes it not complain on compilation if there's no Serial1
  #define BLUEFRUIT_HWSERIAL_NAME      Serial1
#endif


// SHARED UART SETTINGS
// ----------------------------------------------------------------------------------------------
// The following sets the optional Mode pin, its recommended but not required
// ----------------------------------------------------------------------------------------------
#define BLUEFRUIT_UART_MODE_PIN        -1    // Set to -1 if unused


// SHARED SPI SETTINGS
// ----------------------------------------------------------------------------------------------
// The following macros declare the pins to use for HW and SW SPI communication.
// SCK, MISO and MOSI should be connected to the HW SPI pins on the Uno when
// using HW SPI.  This should be used with nRF51822 based Bluefruit LE modules
// that use SPI (Bluefruit LE SPI Friend).
// ----------------------------------------------------------------------------------------------
#define BLUEFRUIT_SPI_CS               10
#define BLUEFRUIT_SPI_IRQ               2
#define BLUEFRUIT_SPI_RST              -1    // Optional but recommended, set to -1 if unused

// SOFTWARE SPI SETTINGS
// ----------------------------------------------------------------------------------------------
// The following macros declare the pins to use for SW SPI communication.
// This should be used with nRF51822 based Bluefruit LE modules that use SPI
// (Bluefruit LE SPI Friend).
```

```
// ---------------------------------------------------------------------------------------------
#define BLUEFRUIT_SPI_SCK            13
#define BLUEFRUIT_SPI_MISO           12
#define BLUEFRUIT_SPI_MOSI           11
```

Adafruit_VL6180X.h
Adafruit_VL6180X.cpp

Code from Adafruit website [39]

Appendix I: References

---

[1] Foot Stress and Blood Flow Sensing Under High-Loading During Hip Surgery. Stryker Sports Medicine, 2017.

[2] Burlingame, Byron. “Operating Room Requirements for 2014 and Beyond.” 2014 FGI Guidelines Update Series, #3, 15 Sept. 2014. https://www.fgiguidelines.org/wp-content/uploads/2015/10/FGI_Update_ORs_140915.pdf

[3] "Hip Arthroscopy." Clohisyhipsurgeon.com. Department of Orthopaedic Surgery, 2017. Web. 15 Oct. 2017. <http://clohisyhipsurgeon.com/treatment-options/hip-arthroscopy>.

[4] Vipa Dua. “Slipped Capital Femoral Epiphysis.” Orthopedic Surgeon in Hartford, June 2016, www.robotichipandknee.com/your-hip/Slipped-Capital-Femoral-Epiphysis.html.

[5] Hip Arthroscopy: Removing Loose Bodies. Retrieved October 13th, 2017. From, https://www.mountnittany.org/articles/healthsheets/3933

[6] Dahlby, Ryan. On-site visit to Stryker. 12 Oct. 2017.

[7] The Mayo Table Technique in Hip Arthroscopy. Arthroscopy Techniques, 5(3), e459–e463.

[8] Hip Distraction System. (n.d.). Retrieved October 06, 2017, from https://www.arthrex.com/hip/hip-distraction-system

[9] Papavasiliou, A. V., & Bardakos, N. V. (2012). Complications of arthroscopic surgery of the hip. Bone & Joint Research, 1(7), 131–144. http://doi.org/10.1302/2046-3758.17.2000108

[10] King JC. Peroneal neuropathy. In: Frontera WR, Silver JK, Rizzo TD, eds. Essentials of Physical Medicine and Rehabilitation: Musculoskeletal disorders, pain and rehabilitation. 2nd ed. Philadelphia, Pa: Saunders Elsevier; 2008:chap 66.

[11] FDA Quality System Regulation, 21 C.F.R. § 820 (2017). https://www.accessdata.fda.gov/scripts/cdrh/cfdocs/cfcfr/CFRSearch.cfm?fr=820.1

[12] “ISO 13485 – Medical Devices.” ISO 13485 Medical Devices, 29 Aug. 2017, www.iso.org/iso-13485-medical-devices.html

[13] “How Pulse Oximeters Work Explained Simply.” *HowEquipmentWorks.com*, Avada, www.howequipmentworks.com/pulse_oximeter/

[14] “Pulse Oximetry.” *Wikipedia*, Wikimedia Foundation, 28 Nov. 2017, en.wikipedia.org/wiki/Pulse_oximetry.

[15] “Withings Pulse – What does Sp02 mean? What is a normal Sp02 level?” https://support.health.nokia.com/hc/en-us/articles/201494667-Withings-Pulse-What-does-SpO2-mean-What-is-a-normal-SpO2-level

[16] PhD, Charles Patrick Davis MD. “Hypoxia vs Hypoxemia: Click for Signs & Causes.” MedicineNet, www.medicinenet.com/hypoxia_and_hypoxemia/article.htm.

[17] “Patent EP1638456A2 - Mobile health and life signs detector.” Google Patents, Google, www.google.com/patents/EP1638456A2

[18] “What is Perfusion Index (PI)” Amperor Direct USA https://www.amperordirect.com/pc/help-pulse-oximeter/z-what-is-pi.html

[19] King, David. Morton, Robert. Bevan, Cliff. “How to use capillary refill time” http://ep.bmj.com/content/99/3/111

[20] Lima, AP. Beelen, P. Bakker, J. “Use of a peripheral perfusion index derived from the pulse oximetry signal as a noninvasive indicator of perfusion.” https://www.ncbi.nlm.nih.gov/pubmed/12072670

[21] Vuksanović, Vesna, et al. “Nonlinear Relationship between Level of Blood Flow and Skin Temperature for Different Dynamics of Temperature Change.” Biophysical Journal, vol. 94, no. 10, 2008, doi:10.1529/biophysj.107.127860.

[22] Nardin, Rachel A., et al. “Foot Temperature in Healthy Individuals.” Journal of the American Podiatric Medical Association, vol. 100, no. 4, 2010, pp. 258–264., doi:10.7547/1000258.

[23] Henning, RJ. Wiener, F. Valdes, S. Weil, MH. “Measurement of toe temperature for assessing the severity of acute circulatory failure.” https://www.ncbi.nlm.nih.gov/pubmed/451819

[24] Li X, Wang, et al. “Measurement of the Ankle-Brachial Index (ABI). DP Indic | Open-i.” *U.S. National Library of Medicine*, National Institutes of Health, 2013, openi.nlm.nih.gov/detailedresult.php?img=PMC3780660_TSWJ2013-185691.001&req=4.

[25] “What is EMG and Nerve Conduction Study?” WebMD. https://www.webmd.com/brain/emg-and-nerve-conduction-study#1

[26] Industries, Adafruit. “Round Force-Sensitive Resistor (FSR).” *Adafruit Industries Blog RSS*, www.adafruit.com/product/166.

[27] “Doppler Ultrasound Exam of Arm or Leg” https://www.healthline.com/health/doppler-ultrasound-exam-of-an-arm-or-leg

[28] “HADECO SMARTLINK PC SOFTWARE FOR DOPPLERS (Compatible with all dopplers except Minidop)” http://www.surgicalsupplies.co.nz/podiatry/Diagnostic/doppler-software/diagnostic-hadeco-software-smartdrop-xt7-doppler-doppler

[29] “HADECO BIDOP BI_DIRECTION WAVEFORM LCD VASCULAR DOPPLER WITH 8MHZ PROBE (Temp and Toe Pressure Accessories Available) http://www.surgicalsupplies.co.nz/podiatry/Diagnostic/Dopplers/diagnostic-hadeco-bidop-bi-direction-waveform-lcd-vascular-doppler-8mhz-probe

[30] laplaser. “Laser Sensors - Triangulation Principle.” YouTube, YouTube, 20 Dec. 2010, www.youtube.com/watch?v=HsHyYB7ObCU.

[31] “LPPS-SL Series Spring Loaded Linear Potentiometer Position Sensor.” *Harold G Schaevitz LLC The Sensor Connection*, 10 Dec. 2016, thesensorconnection.com/custom-linear-position-sensors/linear-position-sensors/lpps-sl-series-spring-loaded-linear-potentiom?st-t=adwords&vt-k=&vt-mt=&vtap=1o1&gclid=CjwKCAiAu4nRBRBKEiwANms5W9AyvQMQLLSa99tIwbybAYwfWpRu7CbllkTU5M4rugySkI0K7lxqORoCnGQQAvD_BwE.

[32] “Introduction to the XBee modules.” Retrieved November 6th, 2017. From, http://docs.digi.com/display/RFKitsCommon/Introduction+to+the+XBee+modules

[33] “How It Works.” Retrieved November 6th, 2017. From https://www.bluetooth.com/what-is-bluetooth-technology/how-it-works

[34] Comparison of Wireless Technologies (NFC - WIFI - Zigbee - Bluetooth - GSM). Retrieved November 6th, 2017. From http://www.fut-electronics.com/wp-content/plugins/fe_downloads/Uploads/Comparison%20of%20Wireless%20Technologies.pdf

[35] Ray, Brian. “Bluetooth Vs. Bluetooth Low Energy: What’s The Difference?” November 01, 2015. *LinkLabs Blog.* https://www.link-labs.com/blog/bluetooth-vs-bluetooth-low-energy

[36] “Using an FSR” https://learn.adafruit.com/force-sensitive-resistor-fsr/using-an-fsr

[37] “Arduino Uno Rev3”. https://store.arduino.cc/usa/arduino-uno-rev3

---

[38] “Adafruit Bluefruit LE SPI Friend – Bluetooth Low Energy (BLE)”. https://www.adafruit.com/product/2633

[39] “Arduino Code”. https://learn.adafruit.com/adafruit-vl6180x-time-of-flight-micro-lidar-distance-sensor-breakout/wiring-and-test