Team Naura
NAVSEA Robosub
Alpha Prototype Results
EE 415
December 4, 2017

Mentors:
Aaron Darnton
Alex Read

Co-mentor:
Daniel Hubert

Faculty Volunteer:
Andy O’fallon

Professor:
Patrick Pedrow

Group Members:
Henderson Reaves
Nicholas Heitman
Luke Riske
Thong Huynh

Alpha Prototype Description

The objective for this semester was to develop a leak detection system for the Robosub club's autonomous submarine, and present an alpha prototype model of the design to our industry mentors, and primary stakeholders involved. Our team followed an iterative design procedure until finally selecting a concept, and constructing the alpha prototype model. Our alpha prototype is defined as a pre-submergent leak detection system, which has been implemented by pulling a vacuum, and monitoring pressure differentials on a sealed vessel. The system is designed to require minimal human interaction, and provides clear and concise results. This scheme is standalone, and completely detached from the submarine while it is underway. Our team is confident this approach can also be useful on relative applications that need to be tested for airtightness.

Summary of Alpha Prototype Session with Industry Mentor

After introduction of ourselves, our alpha prototype model, industry mentors, and primary stakeholders to the attendees of our presentation; we began presenting on the development procedure of our alpha prototype. The following information is a summarization of the semester long iterative design process, and alpha prototype design details, which were covered in the alpha prototype presentation.

The first step in our design procedure revolved around establishment, and organization of our team. A group photo of our senior design team is shown in Figure 1, with corresponding names captioned below. The role of communication liaison for the fall semester was taken on by Nick Heitman. Our team assessed each other's strengths early on, and were able to conclude suitable roles for every member. Henderson Reaves, and Thong Huynh specialize in computer engineering, therefore, taking on tasks related with programming of the microcontroller. Luke Riske, and Nick Heitman are electrical engineers, and focused primarily on hardware design, and testing roles. Organization of tasks for each member were taken further by evaluating each other in networking, communication, research, and technical writing skills. Our team met frequently each week of the semester to update, and assess the iterative design procedure throughout the duration of EE415.

Figure 1. (From left to right): Thong Huynh, Henderson Reaves, Nick Heitman, and Luke Riske

It was crucial to develop an affirmative understanding of the design problem before defining goals, and specifications for our project. The abstract for the "Robosub" design problem influenced our team to interview the mentor's from NAVSEA, and club members from Robosub to develop project needs. The summary of the problem also gave a clear indication that a pre-submersion, and internal leak detection system would be the primary focus of our design procedures. After consulting with our clients, it was clear that a water breach could damage thousands of dollars worth of electronics within the autonomous submarine, and their current method of leak detection was inefficient. Our team was able to extract needs for both systems through discussions with both parties. The information in Table 1 indicates the needs collected from meetings held with our NAVSEA mentors, and Robosub club leaders.

Table 1: Primary Needs for Pre-Submersion, and Internal Leak Detection Systems

Pre-Submersion System Needs	Internal System Needs
Minimal human interaction: Require low maintenance, and user inputs.	Send flag bit when hull is breached: Transfer data to the submarines AI system.
Display accurate and user friendly data: Show clearly whether the system is under passing or failing conditions.	Compatibility with their existing software: Must choose hardware that is programmable with their current software.
Quick results: Display test conditions in an acceptable timeframe.	Output to USB: Transfer data through their available ports.
Standalone from the AUV: Detachable, and portable system.	Meet available power requirements: Use their 12-14V onboard power supply.

The Robosub club suggested that our team keeps both systems separate, and focuses primarily on the pre-submersion system in the fall semester.

Before beginning the design phase, it is critical to work within the needs taken from clients, and stakeholders. In this situation it is the responsibility of the engineers to interpret the needs of the client into target technical specifications that can be designed around. The information contained in Table 2 shows the mapping of our clients' needs into target technical specifications.

Table 2: Needs and Metrics with Ideal Margins

#	Need	Metrics (with ideal margins)
1	External standalone functionality	Unit must have minimal user interfacing input, and low tolerance for data error (1 user input, 10% tolerance)
2	Simple design	System that provides a relatively small footprint, and low maintenance (20 square centimeters)
3	Pre-submergence check in timely manner	Check measurements, and conduct tests within a short period of time (Within 5 minutes)
4	Meet available power specifications	Allow system to rely on independent power supply, without wall connection (12V-16V battery supply)
5	Sampling time per measurement	Sample measurements at a fast pace (at least every 2 seconds)
6	Accounting for changes of temperature that could be caused by internal electronics	Allow for fluctuations of temperature (2 degrees celsius)

Once, the target technical specifications were established, we had to consider the broader impacts that our design would have on surrounding systems, and the users of our design. To analyze the broader impacts we utilized the risk management technique, which was introduced by our instructor. There are 4 steps that must be followed to use this technique, which includes identifying risks, evaluating risks, making decisions, and reviewing the decisions. These steps pertain to our system design as follows in table 3.

Table 3: Risk Management technique for impact analysis

Step 1: Identify Risks	Step 2: Evaluate Risks	Step 3: Make Decision	Step 4: Review Decisions
Vacuum may drop the pressure too low	Loss: If the vacuum is overdrawn the test case could implode Probability: Low	Because it would take a very large pressure to implode the cases we moved forward with this decision	This decision was sound because controls restricting the vacuum were implemented
Board mount components may break	Loss: Components would need to be replaced Probability: Medium	Because this risk had a medium probability we continued forward with the decision but with extreme caution	This decision was questionable because we broke a leg off of the pressure sensor. Even though we fixed it.
Power supply could provide too much to the wrong place	Loss: Could cause component and system failure Probability: Low	Because we put protections in place to prevent this risk we moved forward with this decision	This decision was sound because preventative measures were taken
Numerous connections could present vulnerabilities	Loss: Could give inaccurate results Probability: Low	Because we could account for this in our tests we decided to move forward with this	This decision was sound because preventative measure were taken.

As indicated by Table 3, we were able to use the Risk Management technique to analyze the broader impacts of our system. Since these impacts were fairly low we proceeded forward with our design.

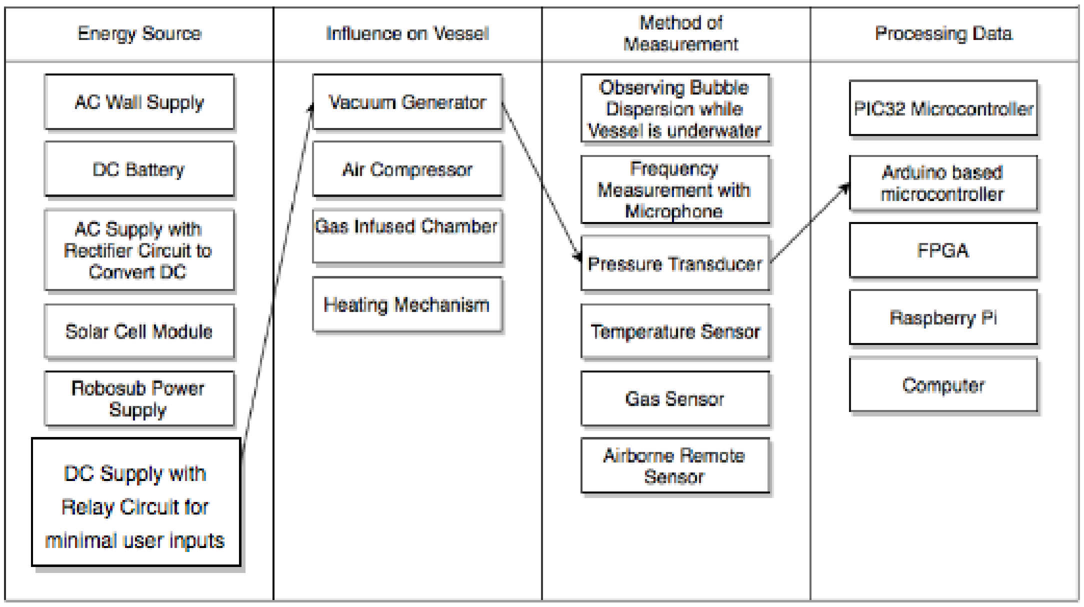
To generate concepts for the pre-submergent leak detection system we searched internally, and externally to develop possible solutions for each subsystem. The external searches included interviewing clients, and stakeholders; researching peer reviewed journal articles, searching patents, and benchmarking related products through the Thomas Register database. Some of the corresponding solutions to each subsystems are shown in Table 4.

Table 4: Concept Generation Table

Subsystems	Solution Concepts
Energy (Power) Supply	● AC wall supply ● DC battery ● AC wall supply with rectifier circuit to convert DC ● Solar cell module ● Robosub submarine power supply ● Relay system between the “influence on vessel,” and “method of measurement subsystems” to centralize user input
Influence on Vessel	● Removal of pressure from the vessel with vacuum generator ● Injection of pressure into the vessel with air compressor ● Injection of heat into the vessel with heating mechanism
Method of Measurement	● Pressure measurement with pressure transducer ● Temperature measurement with temperature sensor ● Gas measurement with gas detector
Data Processing	● PIC32 microcontroller ● Arduino based microcontroller ● FPGA ● Raspberry Pi ● Computer

In the concept selection process for the pre-submergent leak detection system, we evaluated each possible solution generated for each subsystem. Taking the Robosub club’s previous method of leak detection into account, which involved pulling a vacuum, and analyzing pressure differentials with an analog pressure gauge, we decided to follow a similar method by using a pressure sensor controlled with an arduino microcontroller. Since the design needed to be mobile, our team also concluded that a battery supply should be used to power our system. Addition of a relay was also considered to minimize human interaction, and add a safety element to our design. Table 5 shows the concept selection for our design.

Table 5: Concept Selection Table



Energy Source	Influence on Vessel	Method of Measurement	Processing Data
AC Wall Supply	Vacuum Generator	Observing Bubble Dispersion while Vessel is underwater	PIC32 Microcontroller
DC Battery	Air Compressor	Frequency Measurement with Microphone	Arduino based microcontroller
AC Supply with Rectifier Circuit to Convert DC	Gas Infused Chamber	Pressure Transducer	FPGA
Solar Cell Module	Heating Mechanism	Temperature Sensor	Raspberry Pi
Robosub Power Supply		Gas Sensor	Computer
DC Supply with Relay Circuit for minimal user inputs		Airborne Remote Sensor	

The initial hardware design we envisioned is shown in Figure 2, which involved connecting the microcontroller, and pressure sensor between the testing chamber with tubing. Turning on the vacuum generator would bring the combined volume to the same pressure level. Once the correct pressure level is reached the vacuum generator would be turned off by the relay, and the pressure sensor measures the pressure within the test chamber to make sure there is no major pressure deviations in a short timespan.

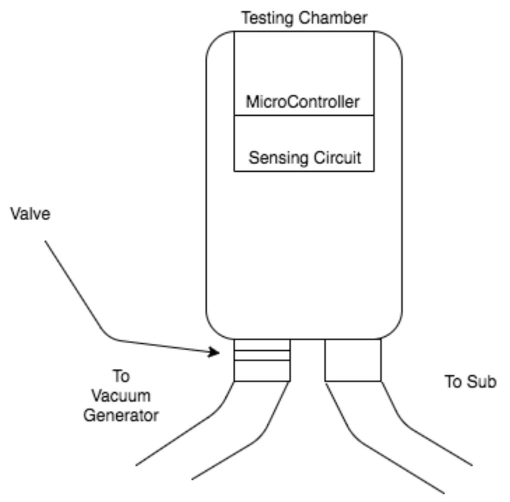


Figure 2. First Concept for the Pre-Submersion System

After looking further into our design, and pondering its abilities, we decided that the model was too bulky for transportion. The design was reconsidered to what is shown in Fig. 3, which consists of a ported pressure sensor connected directly with the vessel, and vacuum through tubing. This not only simplified the design, but allowed for the test chamber to simulate the submarine for the alpha prototype.

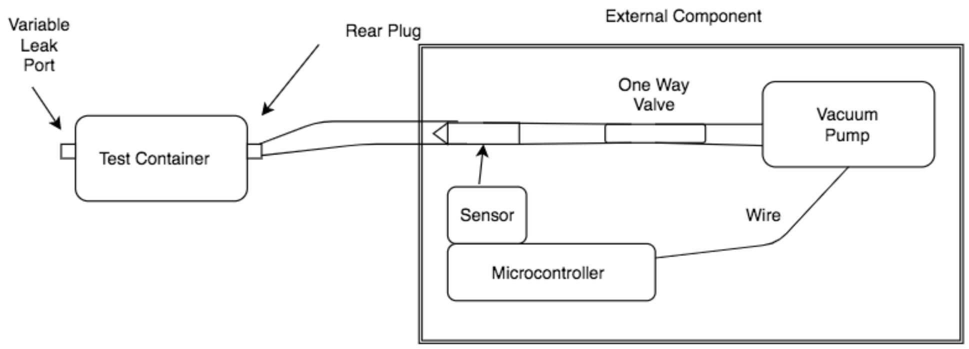


Figure 3. Second Concept for the Pre-Submersion System

Before our design could be built, we were obligated to contact the EHS regarding the safety of pulling a vacuum on a sealed vessel. We were assigned to complete a job hazard assessment EHS assistant director Shawn Ringo. The assessment was meant to identify potential hazards, appropriate controls, and personal protective equipment. Shortly after submission we were informed that our job hazard assessment was acceptable for the design.

The Arduino UNO microcontroller was borrowed from the Robosub club for this design, and is used for interfacing each of the subsystems within the pre-submergent leak detection system. The UNO serves as a placeholder in our alpha prototype for a printed circuit board in the future. After extensive research we determined the HSC differential pressure sensor with SPI protocol would be appropriate for our task. The particular HSC sensor we chose references atmospheric pressure, and uses a ported hose connection to analyze pressure differences from the referral. We were able to utilize a 12V vacuum generator, and two 6V batteries that were in the Robosub clubs inventory. The test capsule used in the alpha prototype presentation is a recycled battery case previously used on the autonomous submarine.

The schematic in Figure 4 shows an overview of the alpha prototype design, which was created with the Fritzing software application. The schematic is rough, in the sense of correctness of specific components, and wire coloring, but it is a good representation of the electronics used in the design. Four subsystems are controlled by the Arduino UNO microcontroller including the LCD display, pressure sensor, BJT switch, and relay coil. The pressure sensor shown in the schematic comes from a different family of sensors, but utilizes the same pin layout, and SPI protocol of our HSC sensor. It should also be noted that the potentiometer used in the schematic is much a larger representation than the$10k\Omega$ sliding potentiometer used in the physical design.

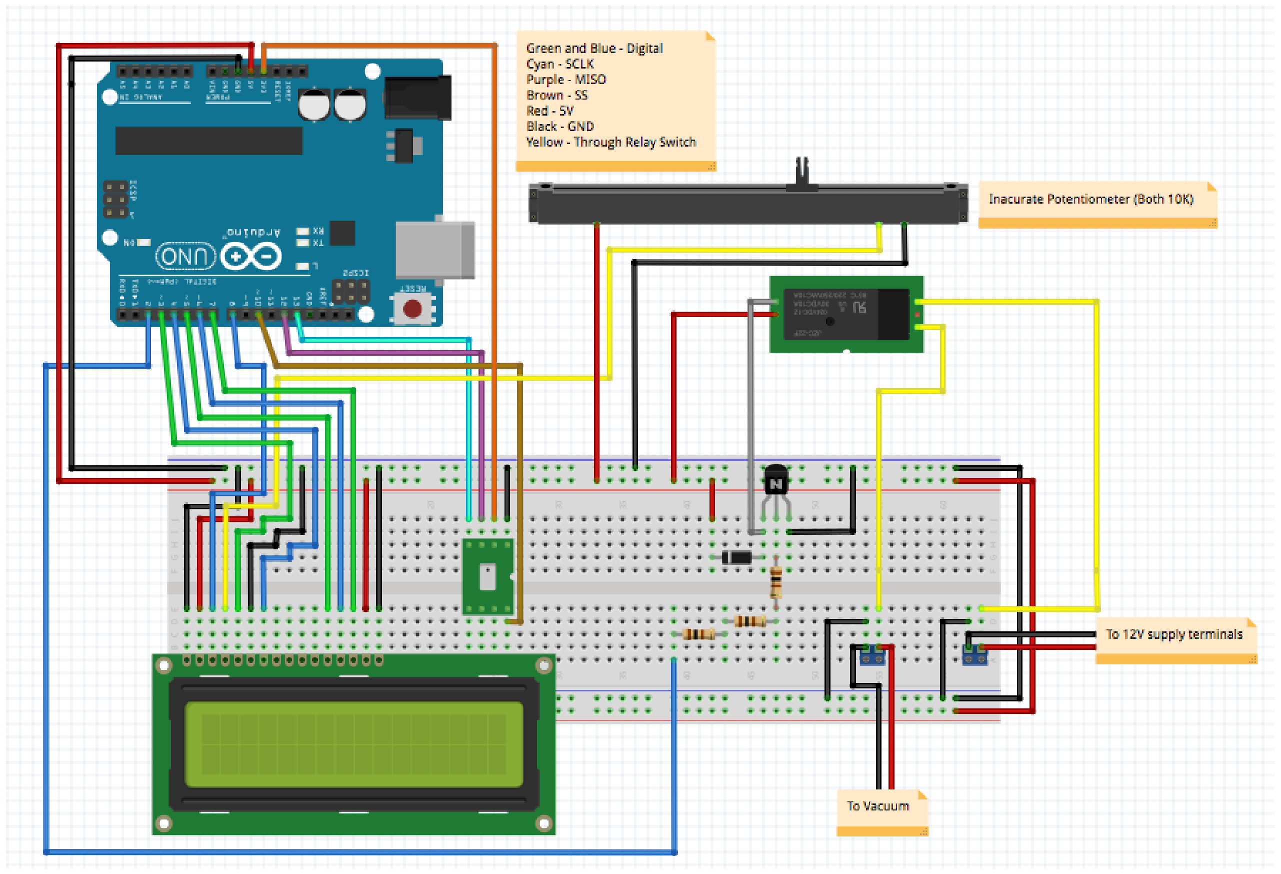


Figure 4. Electrical Schematic of the Alpha Prototype Design

The circuit in Figure 5 shows the relay, which is controlled by a bipolar junction transistor (BJT). The relay coil is connected to the 5V Vcc pin on the Arduino UNO, and is capable of 200mA of current. The collector of the BJT is connected to the opposite polarity of the relay coil, and is used as a switch to control the current flow. The base of the BJT is connected to a 5V digital pin on the microcontroller, which is capable of 20mA of current. From the datasheet of the BJT, it was determined that the ratio of collector, and base current needed to be greater than 10 for saturation (Amplifier Transistors [APA], n.d.). Making the assumption that the relay coil has low resistance under DC conditions, a base current of 15mA, and a 0.7V diode voltage drop from the collector to the emitter, the base resistor was solved as 286.7Ω with a simple KVL loop. In the construction of the physical circuit, three 100Ω resistors were placed in series at the base of the BJT. The diode in Fig. 5 is configured for a safe path to allow current to flow when the BJT is turned off, and current is induced by the change in magnetic field from the coil.

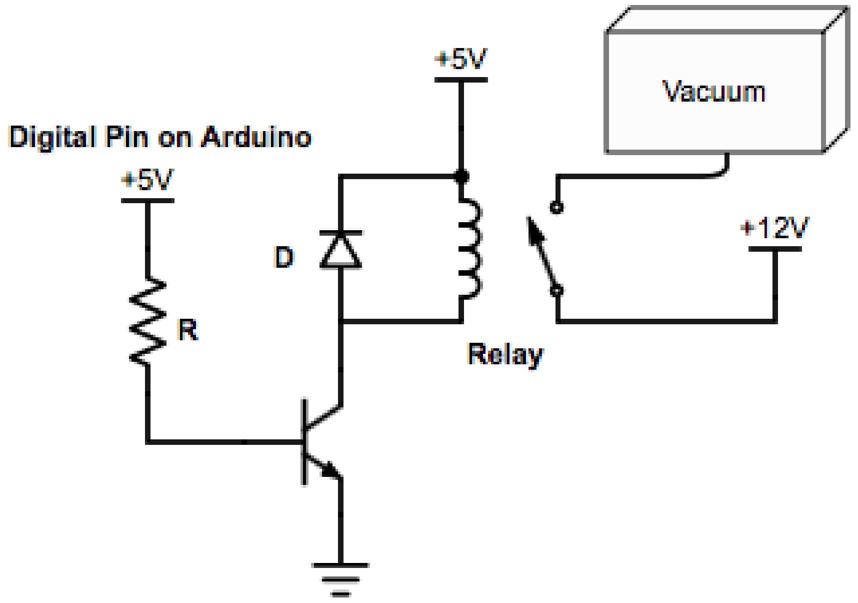


Figure 5. Relay Controlled with a BJT

In order to use our microcontroller (MCU) system, we used Arduino Integrated Development Environment (IDE) software to program the board. Our board has to be Arduino compatible since the robosub club is using Arduino for the Autonomous Underwater Vehicle (AUV). Arduino is an open source platform that is used to build electronic projects. It can be used in different platforms like Windows, Macintosh OSX, and Linux operating systems. We can use programmable circuit board (or microcontroller) with IDE software to upload our code to the physical board. For the pre-submersion test, we use Arduino UNO board and programmed it with C language. A schematic of the Arduino UNO is shown in Figure 6 (What is an Arduino?, 2013).

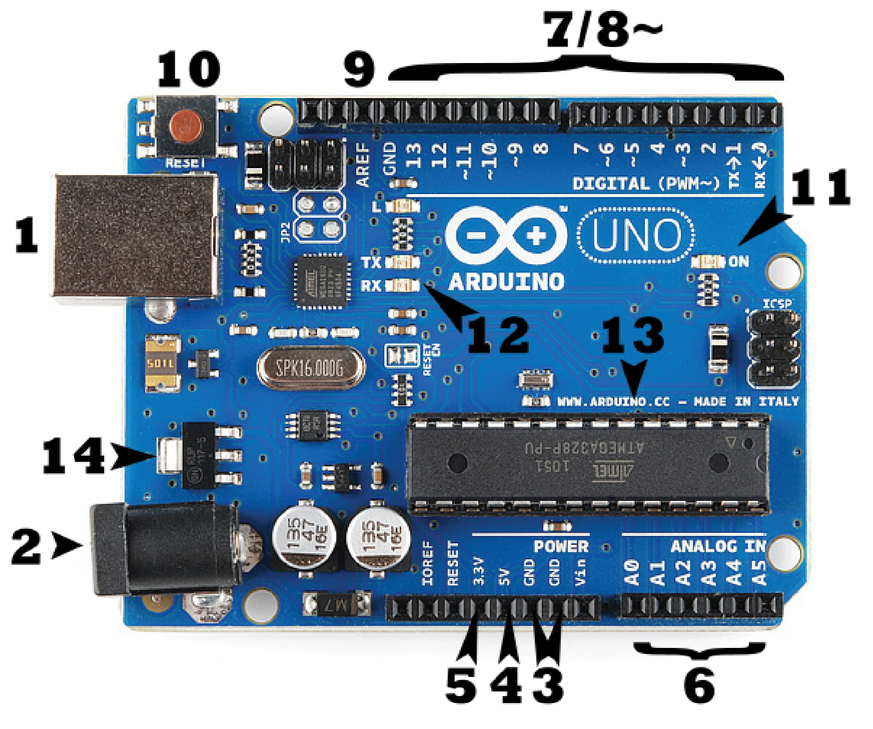


Figure 6. Arduino Uno

The following correspond with the numbers on the Arduino UNO shown in Fig. 6:

- Number 2: Power Jack (Use to connect to independent power source)
- Number 1: USB port
- Number 3: Ground
- Number 4,5: Voltage Supply (3.3V and 5V) for other components.
- Number 6: Analog input signal (Use for Analog sensor).
- Number 7: Digital input/output signal
- Number 8: Pulse-width modulation (PWM) input. It can use for digital pin, but every pin with (~) before them, they can also be used for PWM input signal
- Number 9: Analog Reference. It is used to set external reference voltage (0 - 5V)
- Number 10: Reset button. It is used to reset the board.
- Number 11: Power LED. It is used to indicate that the board is On or Off

- Number 12: TX (transmitter) and RX (Receiver) LED. Indicating the communication between RX and TX.
- Number 13: It is Integrated Circuit (IC). It is a AVR microcontroller and a main controller of the Arduino board. The microcontroller is designed by Atmel company. Different Arduino board will have different microcontroller.

In addition, we use Serial Peripheral Interface (SPI) bus system for the communication between our MCU and sensor. By using the SPI bus system, we can avoid problems with the system clock since the MCU, and sensor have their own clock rates. In the SPI bus system the clock signal, and data signal are sent from the MCU to the sensor separately. This is so the sensor will be able to receive correct data from the MCU since they are running on the same clock system. Moreover, there are two modes for the SPI bus system. The first mode is full-duplex mode, where the MCU receive, and send data to the sensor. The other is half-duplex mode, which is shown in Figure 7. In this mode, the MCU can only receive the data from sensor. The UNO board has a support library for SPI system (A Brief Introduction to the Serial Peripheral Interface (SPI) [APA], n.d.). The half-duplex mode was used for our project, which allows two outputs from the MCU, and one input from the sensor. The first output is Serial Clock (SCLK). The SCLK is the clock signal which is generated by the MCU and it is for the data transmission. The second output is Slave Select (SS) which is used to activate the sensor. The input MISO (Master In/Slave Out) is the sensor line for sending data to Microcontroller (SPI Communication with Honeywell Digital Output Pressure Sensors, 2012).

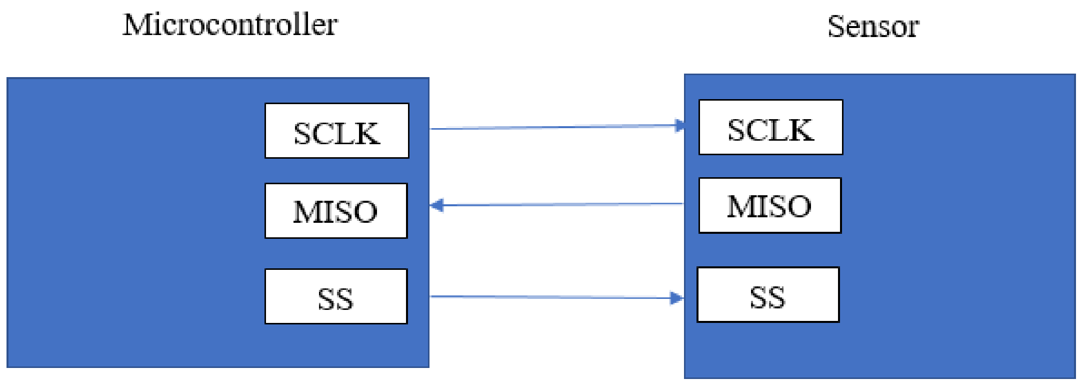


Figure 7. Half Duplex Mode of the SPI Bus System

Moreover, the MCU will use the Slave Select (SS) line to activate the sensor, and it will also generate the clock signal (SCK) for the sensor. Then the sensor will send data back to the

MCU based on each clock pulse from MCU. In addition, the connection pins on the Arduino UNO board for SCLK, MISO, and SS line are 13, 12, 10 respectively. Thus, in order to activate the sensor, we set pin 10 to LOW by writing:

```
uint8_t SS_pin = 10;    // Assign pin for SS line
pinMode(SS_pin, OUTPUT); // Initialize the SS pin as output
digitalWrite(SS_pin, LOW);   // Set the digital SS pin low
```

When the pin of SS line is activated, the MCU will generate a clock signal and send it to sensor via pin 13. When the sensor receive clock signal from MCU, it will send back the pressure data via pin 12. At this point, we wrote a function to read the data from sensor:

```
int16_t Sensor_reading (int8_t SS_pin)
{
SPI.beginTransaction(SPISettings(800000, MSBFIRST, SPI_MODE1)); // 800KHz
digitalWrite(SS_pin, LOW);

int Byte1 = SPI.transfer(0x00);  // Read first pressure sensor byte
int Byte2 = SPI.transfer(0x00);  // Read second pressure sensor byte

 digitalWrite(SS_pin, HIGH); // Deactivate sensor by setting SS line high
 SPI.endTransaction();

int16_t sensor_output = Byte1 << 8 | Byte2; //Required for shifting first byte left
return sensor_output; //Return output pressure reading
}
```

The HSC differential sensor transmit up to 4 bytes of data. The first two bytes contain pressure data, and the last two bytes is optional for temperature. We only use the first two bytes of the sensor in this project. The data read the sensor is shifted in Most Significant Bit (MSB), and it is captured at falling edge of clock pulse. Thus, the output data will be sent to the MCU at rising edge of clock pulse, and the clock phase (CPHA) is 1. The value of clock polarity (CPOL) of our sensor is zero since the clock is at idle when low (SPI Communication with Honeywell Digital Output Pressure Sensors, 2012). Also, the clock frequency of our sensor is from 50 to 800 kHz (SPI Communication with Honeywell Digital Output Pressure Sensors, 2012). Thus, we initialized the sensor settings for the SPI system before activating the sensor to read data with the following code:

```
SPI.beginTransaction(SPISettings(800000, MSBFIRST, SPI_MODE1)); //800KHz
```

The “SPI_MODE1” is used to set the mode of our sensor in SPI library system based on the CPHA and CPOL.

Moreover, We used pin 3, 4, 5, 6, 7, 8 to display the digital output on our LCD screen. The <LiquidCrystal.h> library was used to interface our 16x2 LCD screen. To use the LCD screen on the UNO board we just needed to declare the pin numbers used to connect the LCD in our code, and use the “lcd.print” command to display the output on the screen. We also use “lcd.setCursor(x,y)” to set the position of the output on the LCD screen.

In addition, We used pin 2 on UNO board to control the BJT transistor that controls the relay for our vacuum. We physically connected the base of the BJT to pin 2 on the UNO board, and set pin 2 as an output pin by using “pinMode(2, OUTPUT);” command. After that, we use “digitalWrite(2, HIGH);” command to set the pin to high, and allow for current to flow through the relay coil. Finally, we applied an algorithm to control the relay when we reached a pressure 0.9ATM. We also used the Arduino Serial Plotter to graph our result in real time. The Serial Plotter is a built in application, and it takes incoming data via USB port and graph them on X/Y axis. An example of the Serial Plotter is shown in Figure 8. The Y scale is generated based on our final valued and the X axis has 500 points and each tick is equal to one “Serial.println()” command (New Arduino Serial Plotter, 2015).

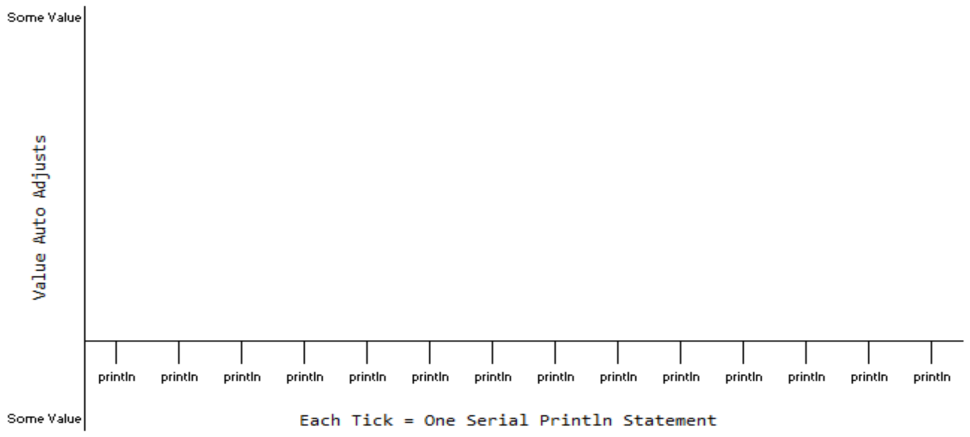


Figure 8. Arduino Serial Plotter

Following the construction of the alpha prototype, we began conducting tests to make sure our design functioned properly. These tests consisted of five trials over a duration of two minutes each. The pressure was taken down to an initial stage of 0.9 ATM, which was approved by our contact at the EHS, and then fluctuations of the pressure in our system were observed. The results of these tests are summarized in Table 4.

Table 4: Pre-submergent leak detection operation test results

Time	Initial Pressure	Final Pressure	Difference
2 minutes	0.9 ATM	0.92 ATM	0.02 ATM
2 minutes	0.91 ATM	0.92 ATM	0.01 ATM
2 minutes	0.9 ATM	0.92 ATM	0.02 ATM
2 minutes	0.91 ATM	0.93 ATM	0.02 ATM
2 minutes	0.91 ATM	0.92 ATM	0.01 ATM

Because the average difference was 0.016 ATM, and the standard deviation was 0.005477, we were able to consider a passing condition of less than 0.93 ATM, and a failing condition greater than 0.935 ATM. These conditions were integrated into our code by using a flag variable to indicate whether or not the pressure rose above 0.935 ATM, a delay cycle to make sure that the test runs for 2 minutes while checking if the flag is triggered, and a pass/fail output that indicates on the LCD if the test passed or failed.

Design Modifications Resulting from Alpha Prototype Activities

After troubleshooting hardware, software, and test logic, our alpha prototype performed as anticipated. However, once we presented our design to our clients and stakeholders, we were informed of some changes that we could make to benefit its functionality. One example would be dropping the pressure down to 0.5 ATM instead of our approved value of 0.9 ATM because this corresponds with the pressure that the sealed vessel would experience at 15 feet below water. We didn't consider the pressure level as it relates to depth because the system would be above water when the test is conducted but it makes sense to bring it down to a level that would actually be experienced by the sub. Another modification to our design is finding another way to display the pressure data because of footprint considerations. The LCD that we are using is quite large in comparison to what our PCB will be once it is refined. Therefore if we want to get it as small and as light as possible we should consider a new display method. The final modification that was suggested to us is to use a MOSFET to replace the relay that controls when the vacuum turns on and off as this would also reduce the size of the final system and simplify the design.

Overall, these changes are minor and can be easily implemented. There were no major design changes that we experienced while creating our alpha prototype. This can be attributed to our concept generation and selection stages of our design procedure, and the collaboration we did with the Robosub Club in the final weeks of the semester.

Future Iterative Design Activities (Plans) for EE416

In EE416 we plan to advance the pre-submergent leak detection system into the beta prototype stage. In the following semester we plan to immediately test our design on the largest volume of the Robosub club's AUV. This will conclude if our vacuum generator is powerful enough to keep for our system. We also plan to add an independent supply for the microcontroller, and implement switches to initiate the test. A MOSFET will be considered to replace the relay in the system, and seven segment displays might replace the LCD to condense the footprint of the electronics.

The current alpha prototype performs the desired tasks although, it can be condensed, and organized. We plan on shifting our breadboard design to a PCB, and using KiCad for design, and OSHPark for manufacturing. While the PCB is being manufactured we plan to find a smaller 12V power supply for our vacuum. Having a smaller power supply will decrease the overall size, and provide ease of transportation for the system. We can then begin to design the parcel for transporting the system. We plan to use the ATmega2560 AVR microcontroller to for our PCB, which is the standard on most of the boards produced by the Robosub club. The Random Access Memory (RAM) of ATmega2560 microcontroller has 8 KB (Atmel ATmega640/V-1280/V-1281/V-2560/V-2561/V [APA], n.d.), and should run smoothly with our software.

To complete the onboard leak detection system, we will resume the iterative design process started in the beginning of fall semester. We will meet with our clients to refresh on the concept we plan to implement. We will present our generated concepts, and new concerns that develop. We currently plan on implementing the onboard model with either conductive leads, or a water detector attached at the bottom of the AUV. The system will send a signal from our microcontroller to the submarine’s CPU if water is detected. It is crucial that we work on this task in parallel with the pre-submersion system to ensure both systems are complete by the end of EE416.

Discussion

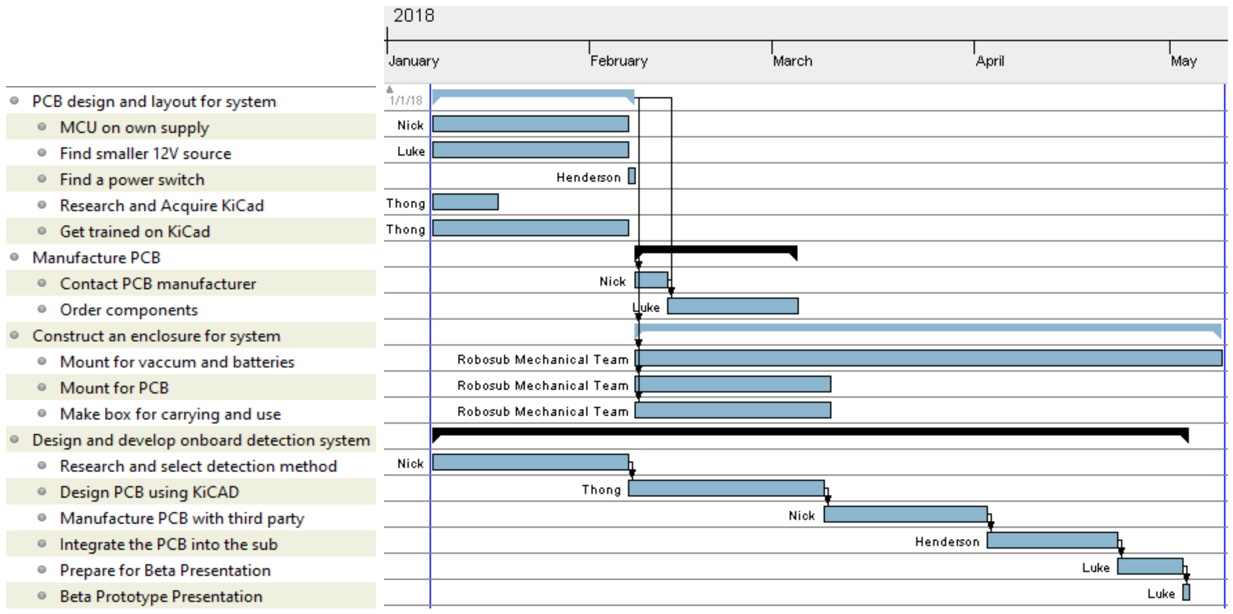
Our team believes we were successful in creating a physical alpha prototype that is representative of what our instructor and clients approved. Although there is need for optimization, as is in most alpha prototypes, we feel that we have made good progress towards our goal of designing a system to solve our client's problem. Because of this achievement, we think that we have progressed as a strong team, and were able to utilize everyone’s strengths to our advantage. We also involved our industry mentors, faculty advisor, and the robosub club to help us get to this point. It is a combination of these factors that will continue to push us towards our goals in EE416.

Conclusion

In conclusion, the demonstration of the pre-submersion alpha prototype system was a success. The design successfully measured the internal pressure of a sealed vessel, and identified whether leak was present. The alpha prototype also plotted live pressure differentials, and displayed the testing state on an LCD screen. We designed this system to be standalone, and require minimal user input. Despite a few components, and parameters that will be tweaked, the design met the desired expectations. We have a plan for advancing our alpha prototype design, as well as our onboard system in the future. Our design team will be focused on completing the necessary tasks immediately to remain on schedule in the following semester.

Appendix

Gantt chart for EE416:



Bibliography:

What is an Arduino? (2013, February 26). Retrieved December 08, 2017, from https://learn.sparkfun.com/tutorials/what-is-an-arduino

A Brief Introduction to the Serial Peripheral Interface (SPI). (n.d.). Retrieved December 8, 2017, from https://www.arduino.cc/en/Reference/SPI

SPI Communication with Honeywell Digital Output Pressure Sensors. (2012, May). Retrieved December 8, 2017, from https://sensing.honeywell.com/index.php?ci_id=45843

New Arduino Serial Plotter. (2015, November 4). Retrieved December 08, 2017, from https://rheingoldheavy.com/new-arduino-serial-plotter/

Atmel ATmega640/V-1280/V-1281/V-2560/V-2561/V. (n.d). Retrieved December 8, 2017, from http://www.atmel.com/Images/Atmel-2549-8-bit-AVR-Microcontroller-ATmega640-1280-1281-2560-2561_datasheet.pdf

Amplifier Transistors. (n.d). Retrieved December 8, 2017, from https://www.onsemi.com/pub/Collateral/P2N2222A-D.PDF