



Coastal Shallow-Water Data Buoy

By

Abdul Khalique Talpur (1245143)¹ - abdulkhalique11@hotmail.com

Hafsa Mir (1345109)² - hafsamir2@gmail.com

Rayan Isran (1345161)³ - rayan.isran@nixorcollege.edu.pk

Project Supervisor

Dr. Raza Akbar – raza.akbar@szabist.edu.pk

A report submitted in partial fulfillment of the requirements for the degree of
Bachelors of Engineering in Mechatronics

Department of Mechatronics Engineering
Shaheed Zulfiqar Ali Bhutto Institute of Science and Technology, Karachi
July 2017

Contents

<i>Abstract</i>	1
<i>Keywords</i>	1
<i>Acknowledgement</i>	2
<i>List of Tables</i>	3
<i>List of Figures</i>	5
1. <i>Introduction</i>	7
2. <i>Methodology</i>	8
2.1 <i>Design Aspects</i>	8
2.2 <i>Background and Design Motivation</i>	9
2.3 <i>Fiberglass Shell Manufacturing</i>	9
2.3.1 <i>Making the Mold</i>	9
2.3.2 <i>Making the Final Casting/Product</i>	10
2.4 <i>Machining Shortcomings and Defects</i>	10
2.5 <i>Assembly</i>	11
2.6 <i>Sealing</i>	15
2.7 <i>Corrosion</i>	16
2.8 <i>Electronics</i>	16
2.8.1 <i>Choice of Hardware Platform Used</i>	18
2.8.2 <i>Choice of Transmission Module Used</i>	20
2.8.3 <i>Disc Layout and Mounting</i>	22
2.8.4 <i>Battery Specifications</i>	23
2.8.5 <i>Coding</i>	24
3. <i>Results and Discussion</i>	26
3.1 <i>Shortcomings and Errors</i>	27
3.2 <i>Possible Test Improvements</i>	28
4. <i>Conclusion</i>	29
4.1 <i>Proposed Future Improvements</i>	29
APPENDICES	30
Appendix A.....	31
Timeline of Project	31

Appendix B	32
Calculations.....	32
Appendix C	43
<i>Materials</i>	43
Appendix D	47
<i>Simulations</i>	47
Appendix E	57
<i>Fasteners</i>	57
Appendix F	58
<i>Machine Drawings</i>	58
Appendix G	62
<i>Mechanical Seals/O-rings</i>	62
Appendix H	64
<i>Project Expenditure</i>	64
Appendix I	67
<i>Electronics Layout</i>	67
Appendix J	68
<i>Hardware</i>	68
Appendix K	77
<i>Software</i>	77
Appendix L	95
<i>Acceleration Graphs</i>	95
Appendix M	99
<i>References</i>	99

Abstract

In the last several decades oceanographic and ocean engineering data buoy systems understanding and technology has greatly advanced [1]. Data buoys provide real time meteorological and oceanographic data. Traditional measurements have included wind speed and direction, barometric pressure, air temperature and humidity, sea surface temperature, salinity measurements, current profiles and waves. The wave data can be used to assess wave action on the coasts and their effects on fauna and flora. This data buoy will provide wave data in terms of heights and periods in the coastal environment. Long term wave data studies can be made to develop local wave trends.

Keywords

Data Acquisition, Wireless Communication, Data Buoy, Data-Logging and Analysis, Video Analysis, Wave-Energy Conversion, Dual-Purpose

Acknowledgement

We would like to express heartfelt gratitude to our project supervisor Dr. Raza Akbar for supporting us throughout our project in technical, conceptual, analytical and other project related difficulties. He provided us the opportunity to work on this project which made us learn how to plan and carry forward an engineering project in organized manner. We also came to know about so many new things which will help us in our careers ahead. We would also like to thank SZABIST Mechatronics faculty for their support and willingness to spend their valuable time with us. We are thankful to PNSC (Pakistan National Shipping Corporation) for machining our project free of cost and helping us in its assembly. In the end, our biggest thanks to our parents for supporting us and encouraging us to complete our project.

List of Tables

Table 2.1 – Optimal Characteristics for Hardware Platform.....	18
Table 2.2 – Comparison of Hardware Platforms.....	19
Table 2.3 – Comparison of Transmission Modules	21
Table A.1 – Project Timeline	31
Table B.1 – Mass Calculation of Container Parts	32
Table B.2 - Buoyancy Calculations (1)	33
Table B.3 - Buoyancy Calculations (2)	33
Table B.4 – Buoyancy Calculation (3).....	34
Table B.5 – Buoyancy Calculation (4).....	34
Table B.6 - Buoyancy Calculation (5).....	34
Table B.7 – Material Dimensions (1).....	35
Table B.8 – Material Dimensions (2).....	36
Table B.9 – Clamped Plate Calculation	37
Table B.10 - Comparison of Center of Mass (Experimental Vs Software)	39
Table B.11 – Sensitivity Analysis of C.O.M	41
Table C.1 – Material Properties	43
Table C.2 – Stock Dimensions of Material (1)	43
Table C.3 - Stock Dimensions of Material (2).....	44
Table C.4 – Stock Dimensions of Material (3)	44
Table C.5 – Stock Dimensions of Material (4)	45
Table C.6 – Stock Dimensions of Material (5)	46
Table C.7 – Stock Dimensions of Material (6)	46
Table D.1 – Stress Analysis Study (Fixture)	47
Table D.2 – Stress Analysis Study (Applied Load)	48
Table D.3 – Results Obtained Using Different Criterion	49
Table D.4 – Stress Analysis Result (Von Mises Stress)	49
Table D.5 – Stress Analysis Result (F.O.S – Distortion Energy)	50
Table D.6 - Stress Analysis Result (Max Normal Stress).....	51
Table D.7 – Stress Analysis Result (F.O.S – Max Normal Stress).....	52
Table D.8 – Stress Analysis Result (Maximum Shear Stress)	53
Table D.9 – Stress Analysis Result (F.O.S - Max Shear Stress)	54
Table D.10 - Stress Analysis Result (Displacement)	55
Table D.11 – Stress Analysis (Top and Bottom Shell Thickness)	56
Table E.1 – Fasteners Used	57
Table G.1 – O-ring Sizes Used	62
Table H.1 – Stock Material Cost.....	64
Table H.2 – Cost of Electronic Components.....	65
Table H.3 – Total Project Expenses	66
Table J.1 – Arduino Mega 2560 Specifications	73
Table J.2 – LiPo Battery Specifications.....	73

Table J.3 – Copernicus Trimble GPS Specifications.....	74
Table J.4 – XBee Pro S2B Module Specifications	74
Table J.5 – LSM9DS0 Accelerometer Specifications	75
Table J.6 - Component Current Consumption	76
Table L.1 – Heave 1 Acceleration Characteristics	95
Table L.2 – Heave 2 Acceleration Characteristics	96
Table L.3 – Heave 3 Acceleration Characteristics	97
Table L.4 – Heave 4 Acceleration Characteristics	98

List of Figures

Figure 2-1 – Rendered Outer View of Buoy	8
Figure 2-2 – Rendered Inner View of Buoy	8
Figure 2-3 – Wire Frame or Mesh Hemispherical Pattern	9
Figure 2-4 – Hemispherical Portion of the Fiberglass Shell	10
Figure 2-5 – Top View of Port (Showing Bolt Holes for Ring)	10
Figure 2-6 – Distance between Adjacent Discs in Container	12
Figure 2-7 - Container Cap Assembly	13
Figure 2-8 – Bottom Ring and Endcap Assembly	13
Figure 2-9 – Bottom Cover Plate Assembly	14
Figure 2-10 – Top Ring and Endcap Assembly	14
Figure 2-11 – Canopy Assembly	15
Figure 2-12 – Electronics Flow Diagram.....	17
Figure 2-13 – Detailed Electronics Flow Diagram of Buoy Rack	22
Figure 2-14 – Electronics Rack Layout	23
Figure 2-15 – Arduino Warning for Low Voltage	25
Figure 3-1 - Tracker Image Showing the White Strip on the Antenna Blend in with the Background	28
Figure B-1 - Coordinate System	39
Figure B-2 – Container's Center of Mass	40
Figure B-3 – Container's Center of Mass Values.....	40
Figure B-4 – Sensitivity Analysis (Y - Coordinate)	41
Figure B-5 – Sensitivity Analysis (X- Coordinate)	42
Figure F-1 – Top Most Canopy Plate	58
Figure F-2 – Canopy	58
Figure F-3 – Top Cover	59
Figure F-4 – Top Endcap.....	59
Figure F-5 – Top and Bottom Ring	59
Figure F-6 – Container Boundary	59
Figure F-7 – Top Container Cap.....	60
Figure F-8 – Top Disc.....	60
Figure F-9 – Middle Disc.....	60
Figure F-10 – Bottom Disc.....	60
Figure F-11 – Bottom Container Cap	61
Figure F-12 – Bottom Endcap.....	61
Figure F-13 – Bottom Cover	61
Figure F-14 – Fiberglass Shell.....	61
Figure G-1 – O-ring Squeeze Calculation	63
Figure I-1 – Fritzing Electronics Schematic	67
Figure J-1 – Arduino Mega 2560	68
Figure J-2 – LSM9DS0.....	69
Figure J-3 – Trimble GPS & Antenna	69

Figure J-4 – LiPo Battery.....	70
Figure J-5 – 5 Volt Buck Converter.....	70
Figure J-6 – Flashing LED	71
Figure J-7 – XBee Pro Series 2B Module	71
Figure J-8 – Arduino Wireless SD shield.....	72
Figure K-1 – XCTU XBee Module Detection	77
Figure K-2 – XCTU Baud Rate Setting for XBee	78
Figure K-3 – XBee Module Data Transmit Test	78
Figure K-4 – XBee Module Data Receive Test	79
Figure K-5 – Arduino Software used for Coding the Microcontroller.....	80
Figure K-6 – Sample Acceleration Data from the Accelerometer.....	81
Figure K-7 – Tracker Software Used for Data Reduction	82
Figure K-8 – Accelerometer Data Packet Format.....	91
Figure K-9 – Sample Accelerometer Data Packet	91
Figure K-10 – Sample Temperature & Voltage Data Packet	92
Figure K-11 – Arduino Message for SD Card Detected	92
Figure K-12 – Arduino Message for SD Card Not Detected	92
Figure K-13 – Received Packet Data on the Arduino Uno Serial Port.....	93
Figure K-14 – Sample Log File on SD Card.....	94
Figure L-1 – Heave 1 Acceleration Graph.....	95
Figure L-2 – Heave 2 Acceleration Graph.....	96
Figure L-3 – Heave 3 Acceleration Graph.....	97
Figure L-4 – Heave 4 Acceleration Graph.....	98

1. Introduction

Data buoys are generally expensive and only purchased from foreign companies when required for use in Pakistan. Our objective is to design a simple and robust yet cost-effective data buoy using standard commodity electronics. The sensors and associated electronics used in the buoy are relatively cheap and easily accessible.

Karachi is a coastal city and useful, local wave data can help the local community like fisher-folks, and also help in climate impact studies. Furthermore, wave data can provide information of use to coastal protection and beach nourishment efforts. However in Pakistan there is minimal research on such topics. Data buoy technology has greatly advanced in the west and may be under development in Pakistan, but we did not come to know or find out about any organization making such buoys. Reports in mainstream media have suggested that channel buoys are being developed locally but they are intended for navigation purposes only and not for wave data acquisition.

The structure of the buoy has been divided into two significant portions - an inner canister and an outer shell. Both sections have been designed taking into account various points of concern such as leakage, corrosion, weight, acting stresses and factor of safety. The electronic components are packed in the inner canister discs and the canister is sealed from both ends. An outer shell is added to provide the device with sufficient buoyancy force to float. The dimensions of different components and respective materials have been chosen on the basis of weight and buoyancy analysis, as well as location of the components (exposed or hidden to sea water or environment). Lighter materials have been selected for the upper portion while heavier materials have been selected for the lower portion of the structure so that the buoy remains upright and the location of center of mass prevents the buoy from excessive rolling. The components are joined through fasteners so that the buoy may easily be disassembled without damage to the components.

2. Methodology

2.1 Design Aspects

The structure is mainly divided into two major sections namely the inner canister and the outer shell. The outer shell is used to provide the device sufficient buoyancy force to float. Protection through O-rings is provided at each level (plate).

Further protection is provided to the electronics inside the canister through O-rings on the top and bottom of the canister. It is attached to the outer structure through threaded rods.

Three rods are used 120 degrees apart to keep the structure robust. They pass through the canister and are locked from both sides through fasteners. Three discs are placed inside the canister. The electronic components are fixed on them. An accelerometer is placed approximately at the measured center of mass. An acrylic canopy is installed on the top. The antennas stick out of the canopy as they are waterproof. A flashing LED attached to the top blinks at regular intervals, to help indicate the buoys location at night.

A hoop/lip is made part of the shell (see Fig. 2-1) to increase the buoyancy force which helps to keep it in line with the local water surface, when waves strike it. It also allows some flexibility in modifying the buoy hydrodynamics through appropriate attachments. A tire or wooden belt may be fixed over the hoop to increase buoyancy as needed.

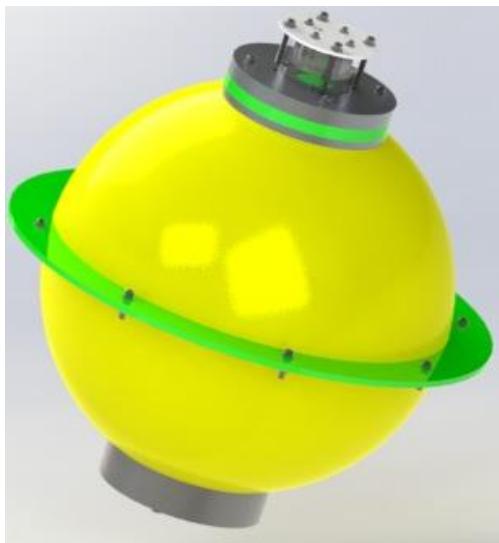


Figure 2-1 – Rendered Outer View of Buoy

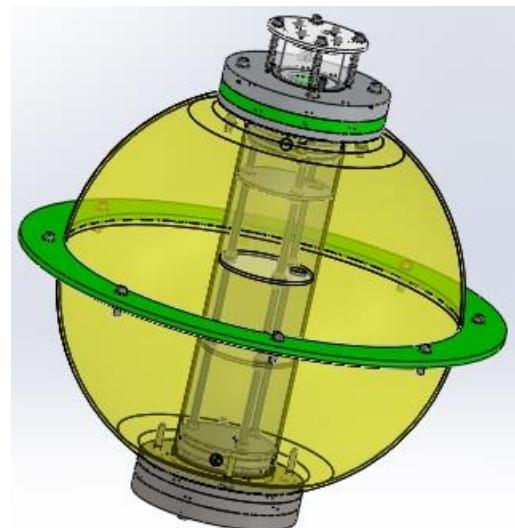


Figure 2-2 – Rendered Inner View of Buoy

2.2 Background and Design Motivation

Why the buoy was made the way it is? The structure or platform has been designed for easy electronics installation, calibration and wave measurements. Fiberglass has been chosen for the outer shell because of its robustness and to keep the buoy safe from corrosion, barnacles and other sea creatures in the sea environment. One major driving force behind designing the buoy was a compact wave-energy conversion device, and so the buoy has been conveniently designed in a way to accommodate power generation from wave motion in the future if need be. Unfortunately, the wave energy generation portion of the project has been set aside due to time and resource constraints.

2.3 Fiberglass Shell Manufacturing

First of all a pattern of the object's shape is made. In order to create a circular or spherical object, a hemispherical pattern is made of metal wire. The hemispherical pattern is then used to make the mold from which several parts can be cast. The mold can be made using different materials like wood, polystyrene/Styrofoam.

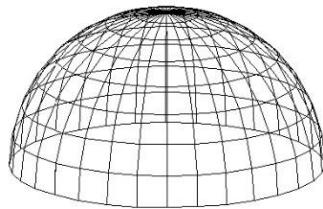


Figure 2-3 – Wire Frame or Mesh Hemispherical Pattern

2.3.1 Making the Mold

After the wire frame hemispherical pattern (flat at the convex side) has been made, a layer of fiberglass fabric is placed over the pattern until it is fully covered and epoxy resin is pasted over it to form the first layer. Similarly, several other layers are placed with epoxy resin as an adhesive between them until the desired thickness of the mold is reached. The inner surface of the mold is finished to the required thickness and dimensions. The radius of the mold is bigger than the final product because the final product is cast inside the mold.

2.3.2 Making the Final Casting/Product

After the hemispherical mold (flat at the convex side) is made, layers of fiberglass (joined together by epoxy resin) are applied inside it to cast the final product. When enough layers are placed and the desired thickness is reached, the mold is left for some time until the cast part inside it dries. A lubricant (oil) is used to easily remove the final casting from the mold. The data buoy shell is made in two parts from the same mold and the parts are then joined together using fiberglass and epoxy at the joining line until an appropriate thickness is formed. The flat sides (see Fig. 2-4) at the top and bottom side of the shell were drilled to form the ports for passing canister (see Fig. 2-5). Effort was made to ensure that the top and bottom ports were aligned to within the needed tolerance.



Figure 2-4 – Hemispherical Portion of the Fiberglass Shell



Figure 2-5 – Top View of Port (Showing Bolt Holes for Ring)

2.4 Machining Shortcomings and Defects

Following manufacturing defects were found:

- The O-ring grooves were slightly off than the required dimensions. The width and depth of the grooves were up to the dimensions. The fillet was slightly larger than needed. The O-ring, however, works properly.
- The lengths of the screws which hold the caps with the container were slightly short.
- Initially, the top container cap was made of High Density Polyethylene (HDPE), but was found to be leaking. This was due to a 1 mm difference from the required dimensions.

This defect could not be ignored as it would have ruined the electronics inside. Therefore, a new container cap was machined. This time we used aluminum because it is easy to machine and chances of machining errors are minimal. We needed 0.1 mm accuracy because dynamic seals are used and they do not work properly if the compression is not sufficient.

- The top end cap was made from HDPE initially but it got bent due to a temperature rise while machining the material. The material should be machined in steps at appropriate time intervals to avoid overheating which deforms the material, especially plastic. Ertalon was used subsequently to make the top endcap as it is less sensitive to temperature compared to HDPE. Stepwise machining was carried out to avoid overheating.
- The threaded rods which join the canister with the outer structure were found to be short. The reason was that the fiber glass shell was slightly larger (about 2 cm) than the required diameter. So, the rods were changed.
- The container was meant to be machined using acrylic due to its lighter weight and transparency. The machined container had the correct length, but the diameters on both ends were off by several millimeters due to poor acrylic construction techniques, rendering it impossible for the caps to sit correctly. An aluminum container later replaced the acrylic one.

2.5 Assembly

The three discs that house the electronics are first placed one on top of the other vertically in a rack formation, supported by rods and secured by nuts on each side. The aluminum disc is placed centrally and acts like a cooling fin. Although the vertical distance between the discs can be customized, it has been chosen to accommodate the length of the cables used. The aluminum disc houses an accelerometer that has been placed at the approximate center of gravity. The center of mass of the container was measured by balancing it on thin wooden blocks, first horizontally and then vertically. The point at which the two axes intersected was marked as the approximate location of the center of mass.

A sufficient rod length must be protruded from both ends for the rods to pass through the plates, and rings above and below the container.

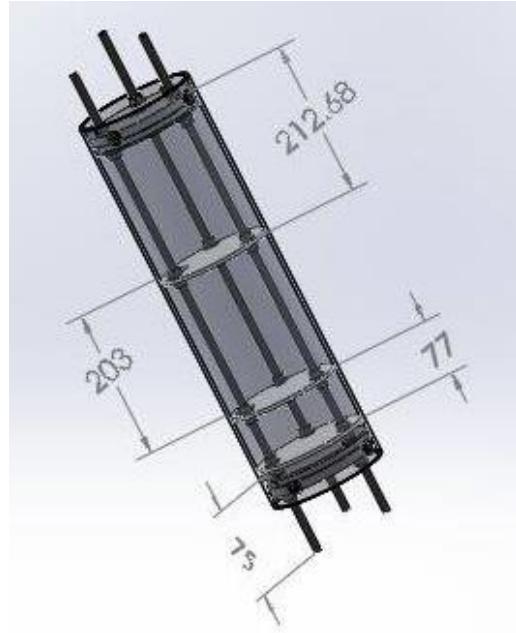


Figure 2-6 – Distance between Adjacent Discs in Container

Once the ‘disc rack’ has been placed inside the container, the container is closed from both ends, with the mild steel cap enclosing the bottom side and the aluminum cap enclosing the top side. For each side, the assembly procedure is as follows:

1. Secure the O-ring on the groove of the cap and make sure the orientation is correct, i.e. the O-ring is placed outwards of the bolt holes in the canister.
2. Using standard silicone grease, lightly grease the boundary of the container where the rings will be placed, i.e. the entire boundary that encompasses the 4 holes. It is recommended to do this twice for extra measure.
3. Repeat process (2) for the O-ring groove on the cap.
4. While holding both sides of the cap using both hands, gently place the cap inside the container. Align the blue mark on the cap with the blue mark on the container. This will ensure correct placement of the holes on the cap with the holes on the container. Slightly adjust to perfectly line them up.
5. With an Allen key, fasten the screws one by one. They need not be very tightly screwed. (see Fig. 2-7)



Figure 2-7 - Container Cap Assembly

It is essential to note that cables protruding from the disc must be taken out from the small hole at the center of the aluminum cap before the cap is fixed onto the container. Place the washers and screw the nuts tightly for each rod on both ends. This concludes the assembly of the container.

The assembly for the bottom side is:

1. O-rings are first greased and placed on the inner and outer grooves of the ring that sits on the bottom side of the shell. This should be done twice for extra measure.
2. The four blind holes of the ring are aligned with the four counter-bore holes of the bottom endcap and then bolted on, whilst also ensuring the rods pass through the equidistant holes at the center of the bottom endcap.
3. As with the ring, O-rings are once again placed on the inner and outer grooves of the endcap and greased all around.

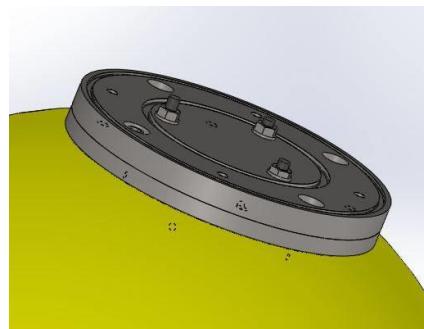


Figure 2-8 – Bottom Ring and Endcap Assembly

4. Another plate, the bottom cover, is then fixed to the endcap.
5. The rods protruding from the container are secured to the three equidistant counter-bored holes at the center of the bottom plate.
6. Bolts are then used on the outer holes to secure the ring to the bottom cover plate.

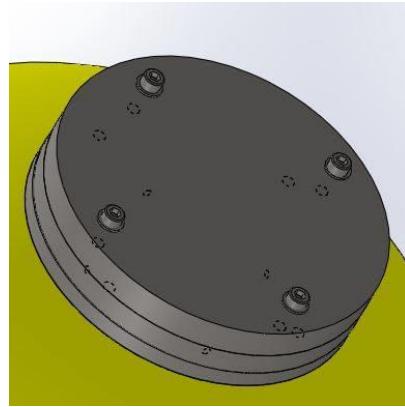


Figure 2-9 – Bottom Cover Plate Assembly

The assembly for the top side is:

1. O-rings are placed on the ring that sits directly on top of the shell and are greased all around, in the same fashion as on the ring that sits beneath the shell.
2. The endcap is then placed on top of ring that sits on the shell such that the holes on the endcap are aligned with those of the ring, whilst also ensuring the three equidistant holes at the center of the endcap pass through each of the three rods.
3. As with the ring, O-rings are placed inside both the inner and outer grooves and greased all around. Bolts are then fastened onto each of the four through holes, effectively securing the ring with the endcap.

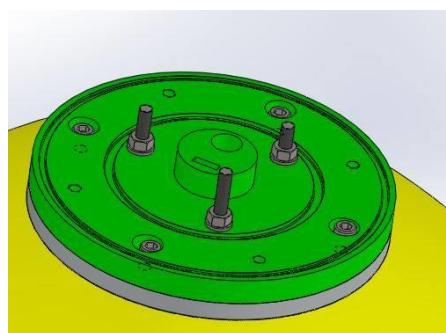


Figure 2-10 – Top Ring and Endcap Assembly

4. The top cover plate is placed on top of the endcap in the same manner, with the end of the rods fastened to the counter-bores on the back side of the top cover plate.
5. The holes on the canopy are aligned with the four inner holes of the top most cover plate using the blue marks on each part as a reference.
6. The top most plate is then bolted onto the top cover plate to secure the canopy in place.

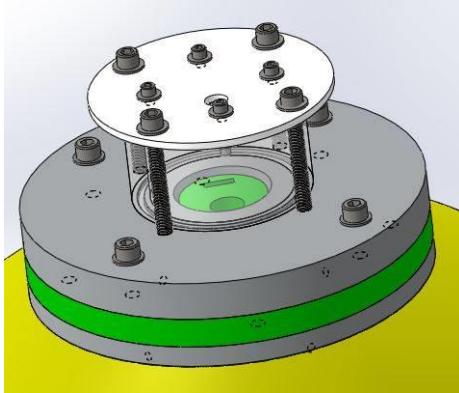


Figure 2-11 – Canopy Assembly

2.6 Sealing

Sealing is essential to make the buoy water tight. O-rings were used to provide sealing. To seal the top side of the shell at which a metal ring sits, an O-ring is placed at the bottom of the ring. It is placed on the ring as machining a groove on fiber glass is relatively challenging. The ring is then attached to the shell using epoxy and RTV as it does not need to be opened for disassembly of the canister. The same procedure is followed for the ring at the bottom.

O-rings are placed at each level where ever there is a leak path. The plate above the ring is the end cap. A pair of O-rings is placed at the top of the ring which surrounds the bolt holes. The bolt holes which join the end cap with the ring are blind i.e. they do not create a leak path. The plate above the end cap is the cover. Another pair of O-rings is placed at the top of the end cap which surrounds the bolt holes. The O-rings are placed in a similar fashion on the plates at the bottom.

A canopy is attached above the top cover using a top most plate. The top most plate is first attached to the canopy through bolts. The top most plate is then joined to the top cover

through bolts. These bolt holes are blind. An O-ring is placed between the canopy and the top cover and another is placed between the top most plate and the canopy.

The canister is sealed using dynamic O-rings at its top and bottom. This provides additional protection to the components inside the canister i.e. if the shell gets damaged, the canister provides sealing.

2.7 Corrosion

The buoy must be corrosion resistant as it sits directly in salt water. The shell is made of fiber glass because it is much less prone to corrosion as compared to steel and aluminum. The plates at the bottom are made of stainless steel 316L which contains 0.03% carbon [13] and is rust resistant. Fasteners used to assemble the device are made of stainless steel.

The rate of corrosion is comparatively much lower for stainless steel but it is not negligible. Thus, cold galvanizing anti seize compound was used to coat the SS plates at the bottom. A Molykote anti-seize compound was used to coat the fasteners to facilitate disassembly and resist corrosion.

Cathodic protection is not necessary to protect the structure and painting can do a reasonable job as the current calculated to protect the structure is only 13 mA. The current required for cathodic protection is calculated using:

$$I = (\text{Surface area of the structure to be protected}) \times (\text{current density of the environment})$$

The surface area of the plates at the bottom is:

$$\text{Surface Area} = 2\pi(0.132)^2 + 2\pi(0.132)(0.06) = 0.16m^2$$

Current density of sea water is 86 mA/m². So, current is found to be 13 mA.

$$I = \frac{0.16(86)}{1000} = 0.013 A$$

2.8 Electronics

The design of the data buoy allows for an easily-configurable rack to be enclosed and sealed within a container, which can house as many electronics as required. One of the preliminary thoughts when designing the buoy was to work out what data would be useful and

how it could be obtained accurately using commodity, student-based electronics. A sensitive, high-resolution accelerometer is used to obtain the wave displacement by integrating the acceleration twice with respect to time. In the event the buoy is operated in a drifting mode, a GPS module would be used to track the location.

A Lithium-Polymer (LiPo) battery powers the Arduino, which in turn provides regulated power to the radio module and sensors. The data acquired from the sensors are sent to the Arduino at specified intervals. The data is then logged to an onboard SD card and simultaneously transmitted through an XBee radio module connected to an antenna. The XBee simply transmits the data to another paired XBee on the receiving end, which is connected to a laptop – in this way the wave data can be accessed remotely in real-time.

We can then send the data over to a logging or spreadsheet software to interpret the results and plot graphs. Eventually, we can plot a graph of the wave data obtained over a long period of time and interpret the wave effects (see Appendix M). A block diagram of the flow is shown below.

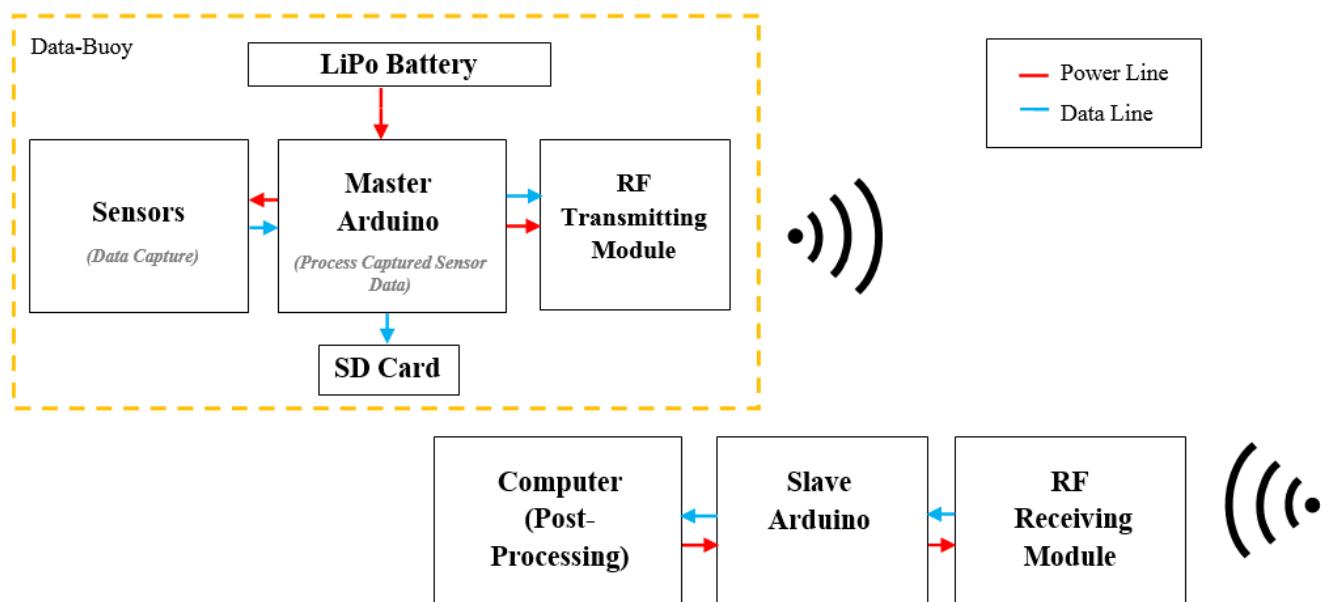


Figure 2-12 – Electronics Flow Diagram

2.8.1 Choice of Hardware Platform Used

Because the buoy was designed with the intention of being a student-based project using inexpensive but effective commodity electronics, the choice of hardware platform was narrowed down to the following four, which are commonly used in design projects:

1. Arduino (unless otherwise mentioned, the Mega 2560 edition)
2. Raspberry Pi
3. Parallax Propeller
4. Intel Edison

To ease picking out a choice, our need considerations were taken into account and the four platforms compared with one another to ascertain which one would be the most suitable choice.

Current consumption	As low as possible to extend battery life.
Support for interfaces	Must have support for I2C and built-in ADC.
Easy to integrate sensors	The appropriate current and voltage requirements on I/O pins must exist.
Cost	As low as possible.
Memory	Enough to hold memory for all variables. If we were to use the camera, we would need a powerful device.
Programming	A language we are familiar with where debugging is also simpler.
Ease of use	As simple as possible. Ability for a layman to understand and edit code if needed using minimum hardware/software.
Size	Small enough to fit on a disc with ample space for sensors/cables if needed.

Table 2.1 – Optimal Characteristics for Hardware Platform

With those details in mind, a simple table was made assessing each characteristic.

Platform →	Arduino (Mega)	Raspberry Pi	Intel Edison	Propeller
Current usage	Low	High	Low	Low
Interface Support	I2C, SPI, ADC built-in	I2C, SPI only	I2C, SPI only	I2C, SPI only
Sensor / Shield Integration	Various shields exist; easy to integrate with components.	-	-	-
Cost	Low	High	High	Low
Memory & Power	Sufficient	Ideal if camera used.	Sufficient	Sufficient
Programming	Arduino Software (similar to C)	Python	C++ with the Eclipse IDE	Spin, Assembly; support for C exists.
Familiarity	High	Medium	Low	Low
Size	10.2 x 5.3 cm	8.5 x 5.6 cm	3.6 x 2.5 cm	5.2 x 1.4 cm
Miscellaneous		Heats up quicker		

Table 2.2 – Comparison of Hardware Platforms

The Raspberry Pi uses an average current almost eight-fold that of the Arduino Mega [2; 3]. This would drain the battery at a much quicker rate, and unless a camera to corroborate wave data was used, which was enlisted as an additional perk and not necessarily a requirement, the high current consumption would make it difficult to justify its usage in the project.

Only the Arduino contains a built-in ADC. Although this is currently only used for the voltage monitoring sensor of the system, ADC support could benefit other sensors in the future, such as a humidity or temperature module. Various shields, such as the Wi-Fi board with an XBee and SD card slot we required, were easily available. The I/O pins on the Arduino also support 5V and enough current required for the accelerometer and GPS module and the strong familiarity with the software makes it a good choice. On the other hand, for the Raspberry Pi, extra hardware such a computer screen, additional keyboard and mouse are required. This would make it difficult for any other programmer to later modify or add code, knowing they would need extra equipment.

The size of each module is moderately small and capable of fitting onto a disc easily. The Intel Edison is the smallest, almost the size of an SD card adapter, delivering great performance but the lack of familiarity with the software and uneasy integration of various sensors such as an XBee made us reluctant towards this choice.

Ultimately, we converged on an Arduino Mega 2560. The Uno possesses low flash memory, which could cause stability problems to occur with the program, hence the preference for the Mega. The extra pins on the Mega could also be used for more sensors in the future. A downside of the Arduino is its limited, on-board 10-bit ADC. Because the accelerometer outputs 16-bit data, the data is invariably sampled at a lower resolution resulting in loss of accuracy.

2.8.2 Choice of Transmission Module Used

A suitable mode of transmission was required to transmit data remotely from the data buoy to a computer. The following modes of transmission were reviewed:

1. XBee ZigBee Transmission
2. Wi-Fi
3. NRF2401 Radio Frequency Modules
4. Bluetooth

Although fairly common these days, Bluetooth was ruled out on the basis that the range is only 100 meters [6] – this is a drawback as the buoy is intended to be used in remote locations in the ocean requiring a lot more than this distance for data transmission.

A comparison chart was made, once again keeping in mind the requirements of the buoy.

Transmission Modules →	XBee	Wi-Fi	NRF2401
Data Rate	250 kbps	54 Mbps – so approximately 25-50 times faster than the NRF2401	Up to 2 Mbps
Power	Requires 2.8 to 3.4V, 40 mA operating current, 63mW output power	3 to 3.6V operating voltage, high power consumption	1.9 to 3.6 V RX: 18 mA TX: 10.5 mA
Range	Up to 1 mile out door range, clear of obstacles	Up to 4.5 km.	40m+ with straight line of visibility; 100m but risks of packet data drop
Cost	High	Low	Low
Pins Required	RX TX VCC GND (4 wires only), convenient	RX, TX, GND and power only (does require voltage divider for TX/RX pins for 3.3V)	Requires 7 I/O pins.
Ease of Use	Simple, documentation and sample codes available	Documentation easily available	Libraries, documentation and sample codes available
Misc.	-	Requires a stable, always active Internet connection	-

Table 2.3 – Comparison of Transmission Modules

A very significant drawback of the Wi-Fi module is the constant need of an Internet connection – this may not be practical in remote locations. A choke in the Internet connection

means no data is received in real-time to the remote computer. While the NRF2401 RF module is cheap and easy to use, the range is a setback.

Despite its high cost, the XBee is a very easy RF solution to integrate. They can be used in transparent mode, in which the XBees identify each other using a Destination Address set in their memory. Using packet formation, data is sent through the TX of one XBee and automatically received by the RX of the destination XBee. With the pro modules, the range of transmission can be as long as 1 mile, a fitting choice for our application. Additionally, The RP-SMA XBees were chosen so that an external antenna of matching impedance could be connected, protruding from the top ring cover to boost signals.

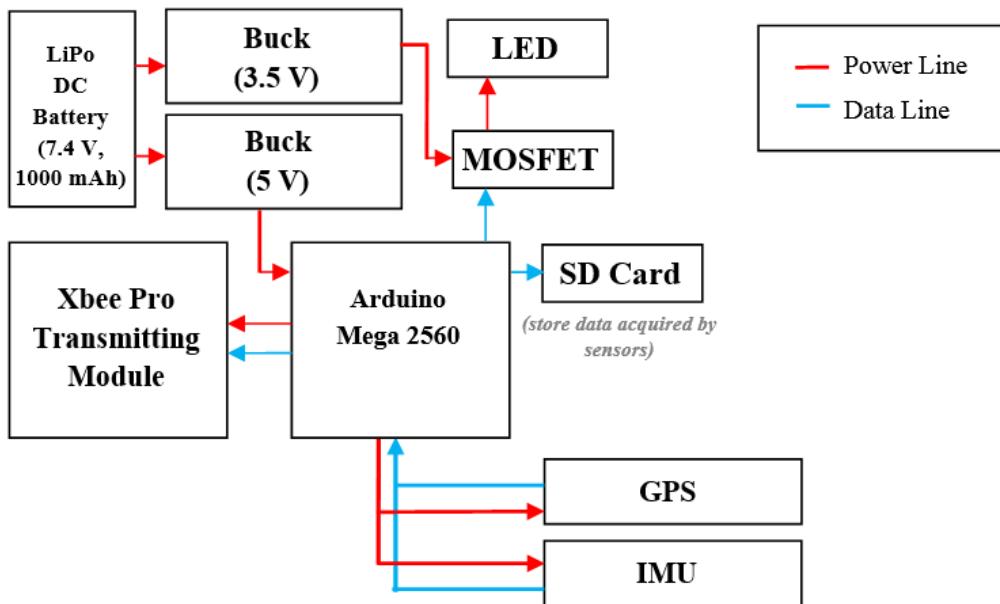


Figure 2-13 – Detailed Electronics Flow Diagram of Buoy Rack

2.8.3 Disc Layout and Mounting

Three discs were chosen as an appropriate number of discs to hold all of the electronics. The bottom most one, bearing most of the weight, would hold the battery and its connecting cables. The top most one holds the microcontroller of choice and the ‘center’ disc houses the accelerometer, which is placed right on the center of the mass, and other sensors if needed including the Raspberry Pi in the event a camera is integrated.

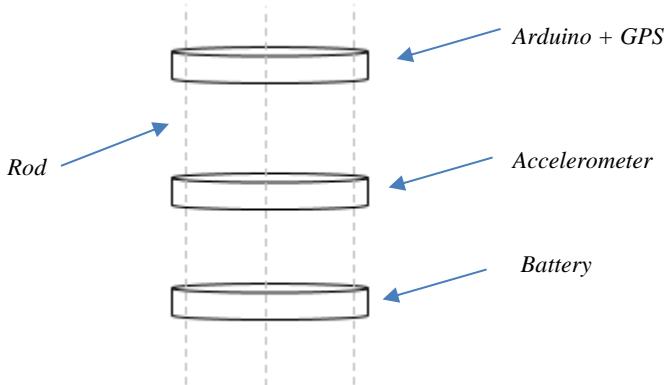


Figure 2-14 – Electronics Rack Layout

A rudimentary sketch created during the initial design phase showing the layout of the electronics is shown in Appendix F and a detailed Fritzing schematic of the current layout is shown in Appendix G.

2.8.4 *Battery Specifications*

Because the data buoy is floating in a remote location and sealed without any built-in sustainable power generation mechanisms, the electronics can only be powered for so long before the battery must be recharged. A single LiPo cell has a nominal rating of 3.7 V, and because some of the electronics require higher ratings such as 5 V, multiple cells have been placed in series. Calculating the total average amount of current consumed is also crucial in order to gauge what battery rating is required. The relationship between the battery capacity and the life of the electronics is given by:

$$\text{Battery Life} = 0.8 * \frac{\text{Battery Capacity (mAh)}}{\text{Current Consumption}}$$

$$\text{Battery Capacity (Ah)} = \frac{\text{Current Consumption} * \text{Battery Life}}{0.8}$$

The efficiency factor is a value close to 1 that takes into account external factors that impact battery life and performance over time, such as the depth of discharge, temperature effects and the cyclic life of the battery. For instance, if we assume an efficiency factor of 0.8, then the required battery capacity to run the on-board electronics for 3 hours is,

$$\text{Battery Capacity (Ah)} = \frac{0.77 * 3}{0.8}$$

which is approximately 2900 mAh, a typical value that two LiPo batteries in series are capable of supplying in a single cycle. A greater capacity battery can power the electronics for a longer period of time. LiPo cells can also be 'packed' in parallel to increase the capacity, at the cost of increased weight and expense. However, because a LiPo battery's weight is small relative to that of the structure and its cost is relatively low, it may be desirable to stack them in parallel in the event current-intensive sensors are installed or the data need to be acquired over a longer period.

2.8.5 *Coding*

The Arduino initializes the accelerometer and the GPS module during its start-up routine. If the LSM9DS0 module is not found, the program waits indefinitely until it is connected. It also looks for an SD card on-board and writes a file titled '**Data.txt**' to it if found. If this file already exists, then data is appended to it rather than overwritten. As of now no error message is generated in the event the SD card has no space.

During the running operation of the program, data is first acquired from the LSM9DS0. This includes the acceleration and gyroscopic data, which is stored and sent in packets. This is printed over the serial monitor at a baud rate of 115,200 which the XBee module picks up and transmits over radio. At the same time, assuming an SD card is connected; data is flashed to it every 1 second.

Every few seconds, an LED mounted on the canopy flashes. After a configurable amount of time, the voltage of the battery and temperature of the container, the latter which is obtained from the LSM9DS0 module, are recorded. By default, the time has been set to 1 minute. A shutdown procedure puts the Arduino in sleep mode when the voltage drops below a certain threshold, which has been set by default to 5.2 V – this being a reasonable threshold to prevent the LiPo battery from falling into an low voltage state, from which it would be difficult to recharge using a LiPo charger. GPS data is also logged once every 5 minutes as a NMEA sentence.

The receiving XBee, already paired with the transmitting XBee, receives and posts the packet. This process is repeated for the next packet and continues indefinitely until communication is ended. When the battery voltage is critical, the following message appears on the serial monitor:

```
{Low voltage detected. Putting Arduino in sleep mode and ending data acquisition..}
```

Figure 2-15 – Arduino Warning for Low Voltage

A full format of the data structure is outlined in Appendix K.

3. Results and Discussion

On the day of the test the following procedure was followed:

1. The electronics rack was first tested on its own, i.e. without being enclosed, and by heaving and rolling it. This was to ensure a last-safety check that data was reliably being transmitted over XBee and being logged to the SD card. This test was approximately 10 minutes long and successful. For simplicity and battery capacity limitations, the flashing LED was discarded.
2. The LiPo battery was then recharged.
3. The rack was placed inside the container. As the container is made out of Aluminum, care was taken to avoid any shorts to ground. Any exposed wires were covered using electrical tape.
4. The buoy was then assembled, following the same steps as outlined in the Assembly section. At each stage of assembly, the power switch was briefly thrown to check data transmission. The switch was then attached to the top most cover plate in a temporary manner which will be fixed in the future.
5. A pool of 0.8 m depth was half-filled with water and the buoy was gently lowered inside.
6. The switch was activated and the first test was a range check. Data was being transmitted to the slave Arduino on the laptop from at least 20 m away in an obstacle-free zone.
7. A camera was placed on a table and remained stationary while yielding a clear, side-ways view of the buoy bobbing up and down. About 5 seconds prior to any data logging, video footage commenced.
8. A count-down was then started followed by an audio cue. At that exact moment, the buoy was heaved once by pushing it from above and left to bob on its own. An approximately time **T_{start}** on the serial monitor of the laptop was noted. For roughly **20 seconds** data was logged.
9. After 20 seconds, data from the serial monitor was copied to a notepad file beginning from **T_{start}** and saved. Video footage was also stopped (see Fig. K-14 in Appendix K for a sample log file).
10. The process was from steps (7) to (9) was repeated thrice. An individual log and video file was saved for each.

For a 20s heave motion, at least 300 acceleration readings were recorded on the serial monitor. Graphs were plotted showing the variation of acceleration with time and are shown in Appendix Z. During the trials, the temperature recorded by the accelerometer was also recorded solely as a safety precaution to ensure the electronics were away from their maximum temperature ratings.

3.1 Shortcomings and Errors

- Because a manual audio cue was used to synchronize the video footage with the accelerometer data oncoming through the serial monitor, there will invariably be an error offset during analysis.
- The data packets on the serial monitor were approximately 60 ms apart, which is a large margin, caused by an extra delay when sending data through the serial port. The time between samples was also not fixed and varied by ± 1 ms.
- For the second trial, the camera was moved at $t = 8.5$ s, rendering the rest of the footage ineffectual to analyze.
- The camera recorded video at a resolution of 240p. This would generate an error during the video analysis as it would be difficult to mark the fiducial point for each frame.
- The intended fiducial mark was a white strip of tape on the XBee antenna, but this camouflaged with the background and as such, another point of reference was used – the red piece of tape enveloping the switch due to its clear color distinction.
- The accelerometer, like any other, is very sensitive to noise. The effect of perturbations can be seen in the graphs in Appendix Z.
- An error in the initial velocity estimate causes a linear position drift even if the acceleration values are unbiased.
- Despite being connected to the Arduino, the GPS module did not log any data for the tests due to loose connections that were detected once the buoy was disassembled.
- Upon inspection, the data on the SD-card file was not logged correctly, missing a handful of packets for reasons currently unknown.



Figure 3-1 - Tracker Image Showing the White Strip on the Antenna Blend in with the Background

3.2 Possible Test Improvements

Many improvements could be made to obtain more reliable readings of the test data, some of which are listed below:

- Use dedicated software to record footage of the camera video alongside the serial monitor so that they are exactly synchronized to mitigate any offset errors.
- Using a higher resolution quality camera with a higher framerate for video analysis.
- Some technique to automatically start logging data before the buoy is heaved.
- Record a section of data prior to setting the buoy in motion.
- Using another accelerometer mounted inside to corroborate the acceleration data.

4. Conclusion

The aim of the project was to develop a robust data buoy that could transmit data reliably in real-time to a receiver that can be several hundred feet away. One of the major goals in the design was to utilize a simple structure and mass-market electronics to keep the cost minimal. Although we were unable to go through with our initial idea of a wave energy generation device, the design of the buoy allows for the electronics rack to be easily replaced with a linear-motion type generator, effectively making the buoy a dual-purpose device. In its current state, we have been able to log and transmit both acceleration and gyroscopic data. Temperature readings from the accelerometer were also recorded. The logging of GPS data was successful but not during tests hitherto, but, this faculty will be developed further.

4.1 Proposed Future Improvements

Although the buoy in its current state and fully functional, it can be further enhanced and several additions can be made, some of which are listed below:

1. Stacking high-end batteries to power the buoy to run for days.
2. Additional sensors can be integrated to collect more oceanographic information, such as water temperature, conductivity and depth.
3. Solar panels for recharging the battery and additional instrumentation may be installed externally.
4. A camera can be installed inside the canopy as another means to measure wave displacement.
5. A GUI to display processed results such as acceleration, velocity and displacement, in real-time.

APPENDICES

Appendix A

Timeline of Project

Activity	Project Timeline												Duration
	August	September	October	November	December	January	February	March	April	May	June	July	
Project Proposal (Report)													
Title Defence (Presentation)													
Preliminary Layout/Design													
Detailed Design													
Procurement Of Electronics													
Procurement Of Material													
Mass Calculation and Buoyancy Studies													
Mechanical Structure Fabrication													
Fiberglass Shell Manufacturing													
Component Study and Testing													
Mid-Year Report and Presentation													
Mid-Year Evaluation													
Conference Paper Introduction													
Assembly of Mechanical Structure													
Leak Check and Buoyancy Test													
Electronics Layout													
Final Version of Report													
Writing Conference Paper													
Testing and Troubleshooting													
FYP Demonstration + Result													

Table A.1 – Project Timeline

Calculations

Appendix B

1. Mass Calculation of Container Parts

S. No	Container Part Name	Total Parts Quantity	Material	Material Density (kg/m ³)	Outer Dia (cm)	Inner Dia (cm)	Height (cm)	Thickness (cm)	Calculated Volume (cm ³)	Mass (kg)	Total Mass (kg)
1	Top Disc	3	Al 5083	2660	15	0	0.5	Height	88	0.235	40.7
2	Middle Disc		Acrylic	1180	15	0	0.5		88	0.104	
3	Bottom Disc		Acrylic	1180	15	0	0.5		88	0.104	
4	Top Cover	1	Al 5083	2660	26.4	6.4	2.6		1340	3.563	
5	Bottom Cover	1	SS 316L	8000	26.4	0	2.5		1370	10.94	
6	Top End Cap	2	Ertalon	1140	26.4	0	4.1		227	2.589	
7	Bottom End Cap		SS 316L	8000	26.4	0	2		1090	8.758	
8	Top Ring	2	Al 5083	2660	26.4	16.4	1.4	Height	480	1.296	40.7
9	Bottom Ring		SS316L	8000	26.4	16.4	1.4		480	3.899	
10	Top Container Cap	2	Al 5083	2660	14.9	0	2.5		440	1.169	
11	Bottom Container Cap		Mild Steel	7850	14.9	0	2.5		430	3.413	
12	Threaded Rod	1	Galv. Iron	7850	0.9	0	60.4		43	0.337	
13	Container Boundary	1	Al 6061	2700	16.2	15.2	56.2		0.5	1380	
14	Canopy Boundary Wall	1	Acrylic	1180	11.8	9.4	5.8		1.2	232	
15	Canopy Roof	1	Acrylic	1180	11.8	0	1.2		130	0.155	
16	Top Most Plate (Canopy)	1	Acrylic	1180	16	0	0.5	Height	100.5	0.119	

Table B.1 – Mass Calculation of Container Parts

2. Buoyancy Calculations

S. No	Values			Calculations Using The Volume Formula Derived From 'Volume of Revolution' or 'Disk' Method			
				(Considering Chopped Sphere) [V = 2*π*((R ²)*H) - (H ³)/3]			
	Radius 'R' (meter)	Chop Length 'L' = R-A (meter)	Shell Thickness 'Th' (meter)	Shell Volume (m ³)	Volume of Shell Boundary/Thickness (m ³)	Density of Fiberglass (kg/m ³)	Mass of Shell Boundary/Thickness (kg)
1	0.32	0.291	0	0.135	0	1460	0
2	0.315		0.005	0.129	0.006		8.47

Table B.2 - Buoyancy Calculations (1)

Mass of Container Parts (kg)	Mass of Shell Boundary (kg)	Mass of Lip/Hoop/Flange around the Shell	Mass of Top and Bottom Part of Shell Boundary (Which is 15 mm in thickness) (kg)	Miscellaneous Mass (Electronics + Other Extra Material If Added) (kg)	Total Buoy Mass or Weight Mass (kg)	Required Buoyancy Mass of Full Shell (Needs to be double of the Total Buoy Mass or Weight Mass)	Force of Gravity 'Fg' or Weight 'W=mg' acting on the Buoy (N)
40.7	8.47	1.31	2.18	10	62.7	125.3	614.8

Table B.3 - Buoyancy Calculations (2)

Shell Volume 'V' (m ³)	Density of Sea Water 'D' (kg/m ³)	Acceleration due to Gravity 'g' (m/s ²)	Force of Buoyancy acting on the Shell (N)	Force of Buoyancy acting on the Bottom Ring (N)	Force of Buoyancy 'Fb' acting on the Bottom Cover (N)	Force of Buoyancy 'Fb' acting on the Bottom Endcap (N)	Total Force of Buoyancy 'Fb' acting on the Buoy (N)	Buoyancy Mass (kg)
Full Shell	1025	9.81	1363.5	4.9	13.8	11.01	1393.2	142
0.1356			681.8	4.9	13.8	11.01	711.4	72.5
Half Shell								
0.0678								

Table B.4 – Buoyancy Calculation (3)

Buoyancy Mass of Full Shell (kg)	Mass of Bottom Ring (kg)	Mass of Bottom Cover (kg)	Mass of Endcap (kg)	Total Buoyancy Mass of Buoy (Considering Full Shell) (kg)
138.9	0.49	1.4	1.12	142

Table B.5 – Buoyancy Calculation (4)

Buoyancy Mass of Half Shell (Which will be under water) (kg)	Mass of Bottom Ring (kg)	Mass of Bottom Cover (kg)	Mass of Endcap (kg)	Total Buoyancy Mass of Buoy (Considering Half Shell) (kg)
69.5	0.49	1.4	1.12	72.5

Table B.6 - Buoyancy Calculation (5)

Since the 'Buoyancy Mass' (142 kg) is about 2.3 times greater than the 'Total Buoy Mass or Weight Mass' (62.7 kg). The condition for the buoy to float in sea water [i.e. $F_b = 2F_g$ or 'Buoyancy Mass' of the complete 'Full Shell' needs to be double the 'Total Buoy Mass or Weight Mass' has been satisfied for a radius of 0.32 m and a thickness of 0.005 m.

'Total Buoy Mass or Weight Mass' needs to be equal to the 'Total Buoyancy Mass (Considering Half Shell)', for the buoy to sit exactly half in and half out of the water. Therefore, we might need to add up an extra mass of $72.5 - 62.7 = 9.85 \sim 10$ kg in the buoy for it to sit half in and half out of the water.

3. Dimensional Calculations of Container Parts:

S. No	Container Part	Qty.	Material	Actual/Finished Outer Dia (mm)	Actual/Finished Inner Dia (mm)	Actual/Finished Thickness (mm)	Required Outer Dia For Machining (mm)	Required Thickness For Machining (mm)
1	Top Disc	1	Al 5083 (1)	150	0	5	160	8
2	Middle Disc	1	Acrylic (1)	150	0	5	160	5
3	Bottom Disc	1	Acrylic (1)	150	0	5	160	5
4	Top Cover	1	Al 5083 (1)	264	64	26	270	30
5	Bottom Cover	1	SS316L (1)	264	0	25	270	30
6	Top Endcap	1	Ertalon (1)	264	0	41.5	270	52
7	Bottom Endcap	1	SS316L (1)	264	0	20	270	25
8	Top Ring	1	Al 5083 (1)	264	164	14.5	270	20
9	Bottom Ring	1	SS316L (1)	264	164	14.5	270	20
10	Top Container Cap	1	Al 5083 (1)	149.65	0	25	160	30
11	Bottom Container Cap	1	Mild Steel (1)	148.8	0	25	160	30
12	Canopy Cover (Top Most Plate)	1	Acrylic (1)	160	0	5	170	5
13	Canopy Roof	1	Acrylic (1)	118	0	12	118	12

Table B.7 – Material Dimensions (1)

S. No	Container Part	Qty	Material	Actual or Finished Outer Dia (mm)	Actual or Finished Inner Dia or Bore (mm)	Actual or Finished Thickness (mm)	Actual Length (mm)	Required Outer Dia For Machining (mm)	Required Inner Dia or Bore For Machining (mm)	Required Thickness For Machining (mm)	Required Length For Machining (mm)	Required Width For Machining (mm)
14	Container Boundary	1	1 of Al 6061	162	152	5	562	162	152	5	572	508.7
15	Canopy Boundary Wall	1	1 of Acrylic	118	94	12	58	118	94	12	68	370.5

Table B.8 – Material Dimensions (2)

4. Plate Stresses

A clamped plate calculation was performed on all plates to determine the maximum deflection when the loads are applied. The results were found to be approximately equal to the simulation results obtained from SolidWorks.

Plate	Material	Yield Strength (S_y) (N/m 2)	Radius (r) (m)	Thickness of Plate 't' (m)	Young's Modulus 'E' (N/m 2)	Poisson Ratio 'v'	Area of Plate 'A' (m 2)	Pressure (atm) (N/m 2)	Radial Stress (Max) (Rmax) (N/m 2)	Deflection (Max) (m)	F.O.S (1) $\frac{S_y}{R_{max}}$	Tangential Stress (Max) (Tmax) (N/m 2)	F.O.S (2) $\frac{S_y}{T_{max}}$
Canopy Roof	Acrylic	4.50E+07	8.00E-02	5.00E-03	3.00E+09	3.50E-01	2.01E-02	1.00E+05	1.92E+07	1.80E-03	2.34E+00	1.30E+07	3.47E+00
Top Endcap	HDPE	3.20E+07	1.32E-01	4.15E-02	1.70E+09	4.10E-01	5.47E-02	1.00E+05	7.59E+05	3.90E-05	4.22E+01	5.35E+05	5.98E+01
Top Container Cap	Al	6.80E+07	7.50E-02	2.50E-02	1.70E+09	4.10E-01	1.76E-02	1.00E+05	6.75E+05	1.86E-05	1.01E+02	4.76E+05	1.43E+02
Bottom Container Cap	MS	2.82E+08	7.50E-02	2.50E-02	2.05E+11	2.90E-01	1.76E-02	1.00E+05	6.75E+05	1.70E-07	4.18E+02	4.35E+05	6.48E+02
Top Ring	Al	6.80E+07	1.32E-01	1.45E-02	7.10E+10	3.30E-01	5.47E-02	1.00E+05	6.22E+06	2.34E-05	1.09E+01	4.13E+06	1.65E+01
Bottom Ring	SS316L	2.92E+08	1.32E-01	1.45E-02	2.00E+11	2.70E-01	5.47E-02	1.00E+05	6.22E+06	8.66E-06	4.70E+01	3.95E+06	7.40E+01
Bottom Endcap	SS316L	2.92E+08	1.32E-01	2.00E-02	2.00E+11	2.70E-01	5.47E-02	1.00E+05	3.27E+06	3.30E-06	8.94E+01	2.07E+06	1.41E+02
Bottom Cover	SS316L	2.92E+08	1.32E-01	2.50E-02	2.00E+11	2.70E-01	5.47E-02	1.00E+05	2.09E+06	1.69E-06	1.40E+02	1.33E+06	2.20E+02
Disc	Al	6.80E+07	7.50E-02	5.00E-03	7.10E+10	3.30E-01	1.76E-02	1.00E+05	1.69E+07	5.96E-05	4.03E+00	1.12E+07	6.06E+00
Disc	Acrylic	4.50E+07	7.50E-02	5.00E-03	3.00E+09	3.50E-01	1.76E-02	1.00E+05	1.69E+07	1.39E-03	2.67E+00	1.14E+07	3.95E+00
Top Cover	Al	6.80E+07	1.32E-01	2.60E-02	7.10E+10	3.30E-01	5.47E-02	1.00E+05	1.93E+06	4.06E-06	3.52E+01	1.29E+06	5.29E+01

Table B.9 – Clamped Plate Calculation

Formulae Used to Calculate Plate Stresses [12]:

Circular plates in the design are bolted at the corners. Hence, the circular clamped plate calculation can be used as an approximation to determine the maximum deflection of the plates when they are uniformly loaded with atmospheric pressure. It was performed to determine the required thicknesses of the plates. The radial stress is the stress which acts in the radial direction of the plate and its maximum occurs at the edge when $r = R$.

$$\text{Radial Stress (maximum)} = \frac{3Pr^2}{4t^2}$$

F.O.S was calculated accordingly for each plate. F.O.S obtained was above 2 for each plate, thus, they were in the safe range.

$$F.O.S (1) = \frac{S_y}{R_{max}} = \frac{\text{Yield Strength}}{\text{Maximum Radial Stress}}$$

The deflection occurs in the same direction as the loading, along the negative y direction. The deflections obtained were found to be in the safe range.

$$\text{Deflection (maximum)} = \frac{3Pr^4(1 - \nu^2)}{16Et^3}$$

The maximum tangential stress is calculated using:

$$\text{Tangential Stress (maximum)} = \frac{3Pr^2(1 + \nu)}{8t^2}$$

F.O.S was calculated accordingly for each plate. F.O.S obtained was above 2 for each plate; hence, they were in the safe range.

$$F.O.S (2) = \frac{S_y}{T_{max}} = \frac{\text{Yield Strength}}{\text{Maximum Tangential Stress}}$$

5. Center of Mass

The center of mass of the canister was calculated. For simplicity, the three discs inside the canister were not included. The center of mass obtained from SolidWorks is found to be approximately equal to the center of mass obtained experimentally which is shown in the table below:

Center of Mass	Y (mm)	X (mm)
Experiment	1	210
SolidWorks	0	204.3

Table B.10 - Comparison of Center of Mass (Experimental Vs Software)

The reference point for measuring was kept the same to compare results. The coordinate system was chosen as the bottom center of the mild steel container cap shown below:

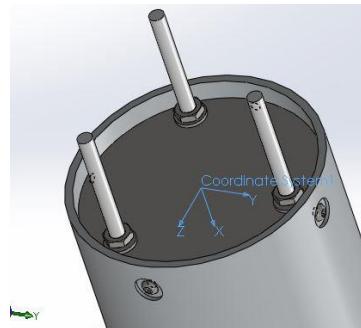


Figure B-1 - Coordinate System

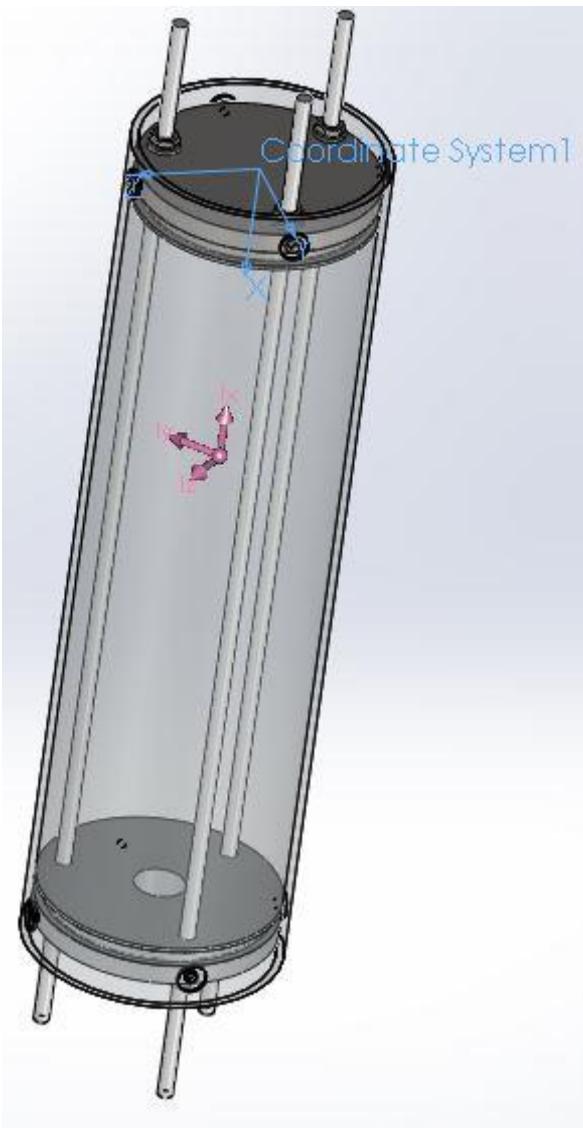


Figure B-2 – Container's Center of Mass

Mass properties of Center of mass (Assembly Configuration - Default)

Output coordinate System: Coordinate System1

Mass = 9.13 kilograms

Volume = 2310475.37 cubic millimeters

Surface area = 735852.00 square millimeters

Center of mass: (millimeters)

X = 204.37

Y = 0.00

Z = -0.00

Principal axes of inertia and principal moments of inertia: (kilograms * square millimeters)

Taken at the center of mass.

I_x = (-1.00, 0.00, 0.00) P_x = 37605.85

I_y = (0.00, -0.71, 0.71) P_y = 409068.74

I_z = (0.00, 0.71, 0.71) P_z = 409068.74

Moments of inertia: (kilograms * square millimeters)

Taken at the center of mass and aligned with the output coordinate system.

I_{xx} = 37605.85 I_{xy} = -0.03 I_{xz} = -0.03

I_{yx} = -0.03 I_{yy} = 409068.74 I_{yz} = 0.00

I_{zx} = -0.03 I_{zy} = 0.00 I_{zz} = 409068.75

Moments of inertia: (kilograms * square millimeters)

Taken at the output coordinate system.

I_{xx} = 37605.85 I_{xy} = -0.02 I_{xz} = -0.24

I_{yx} = -0.02 I_{yy} = 790423.38 I_{yz} = 0.00

I_{zx} = -0.24 I_{zy} = 0.00 I_{zz} = 790423.39

Figure B-3 – Container's Center of Mass Values

A sensitivity analysis was performed in SolidWorks to study the variation in Center of Mass when the distance between the bottom and middle disc changes. The bottom, middle and the top discs were kept at a distance to aid in the assembly. The center of mass was recorded. The middle disc was slightly moved without changing the position of the other discs and center of mass was rerecorded. The same procedure was followed several times and the coordinates of center of mass were recorded noted. The results are shown below:

Distance between Bottom Disc and Middle Disc (mm)	Center of Mass (X) (mm)	Center of Mass (Y) (mm)
77	330.64	-0.07
100	330.08	-0.07
150	328.87	-0.07
175	328.26	-0.07
200	327.65	-0.07

Table B.11 – Sensitivity Analysis of C.O.M

The Center of mass obtained proves to be almost insensitive to movement of the middle disc as the Y – Coordinate remains the same and X – Coordinate changes a few millimeters for a change of about 12 cm of the middle disc. The graphs of both coordinates are shown (see Fig. B-4 and B-5):

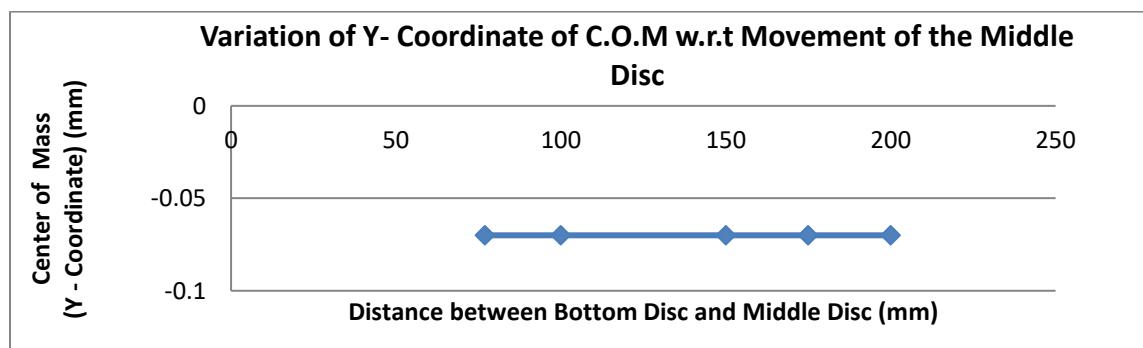


Figure B-4 – Sensitivity Analysis (Y - Coordinate)

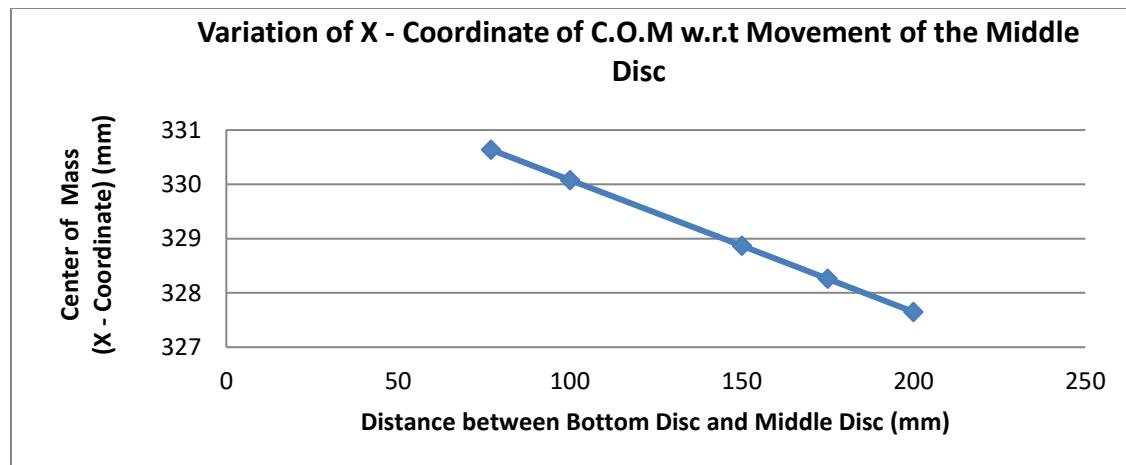


Figure B-5 – Sensitivity Analysis (X- Coordinate)

Appendix C

Materials

1. Properties

S. No	Material	Density	Tensile Strength (MPa)	Fracture Toughness [MPa.(m ^{1/2})]	Melting Point (K)	Melting Point (°C)
1	SS-316L	8000 kg/m ³	480 - 620	112 – 278	1648 - 1673	1375 - 1400
2	Al-5083	2660 kg/m ³	275 - 350	43	864 - 911	591 - 638
3	Al-6061	2700 kg/m ³	276 - 324	29	855 - 925	582 - 652
4	Mild Steel	7850 kg/m ³	370	70 – 80	1755	1482
5	Ertalon	1140 kg/m ³	90 - 165	2.22 - 5.62	563	290
6	Acrylic	1180 kg/m ³	48.3 - 79.6	0.7 - 1.6	513 - 518	240 - 245
7	Fiberglass	1522.4 kg/m ³	200 - 206.8	8.5 - 9	1394.15	1121

Table C.1 – Material Properties [11]

2. Stock Dimensions and Cost

Stainless Steel 316L - Available in Machine Faced/Finished Flanges/Disks (Nanakwara, Paan Mandi, Karachi)												
S. No	Part Name	Required Area For Machining (mm)	Available Area (mm)	Required Finish Thickness (mm)	Available Thickness (mm)	Required Grade	Available Grade	Shop Name	Mass of Available Size (kg)	Cost/kg	Cost (PKR)	Total Cost (PKR)
1	Bottom Cover	270 x 270	270 x 270	25	27	SS 316L	SS 316L	Hatimi Steel Traders	12.5	650 rupees/kg	8,125	23,400
2	Bottom Endcap	270 x 270	270 x 270	20	27				12.5	650 rupees/kg	8,125	
3	Bottom Ring	270 x 270	270 x 270	14.5	23				11	650 rupees/kg	7,150	

Table C.2 – Stock Dimensions of Material (1)

Aluminum Alloy 5083 - Available In Flat Sheets (Nanakwara, Paan Mandi, Karachi)

Part Name	Required Area For Machining (mm)	Available Area (mm)	Required Thickness For Machining (mm)	Available Thickness (mm)	Required Grade	Available Grade	Shop Name	Mass of Available Size (kg)	Cost/kg	Cost (PKR)	Cutting Charges (PKR)	Total Cost (PKR)
Top Disc	160 x 160	160 x 160	8	10	Al 5083	Al 5083	Zulfiqar Traders	0.7	500 rupees/kg	350	200	6,000
Top Cover	270 x 270	270 x 270	32	30				5.9	500 rupees/kg	2950		
Top/Upper Ring	270 x 270	270 x 270	20	25				5	500 rupees/kg	2500		
Container Cap (Top)	160 x 160	160 x 160	30	30				2.04	550 rupees/kg	1122	100	1222

Table C.3 - Stock Dimensions of Material (2)

Mild Steel - Available In Flat Sheets (Condition: New) (Old Haji Camp Road, KMC Workshop, Karachi)											
S. No	Part Name	Required Area For Machining (mm)	Available Area (mm)	Required Thickness For Machining (mm)	Available Thickness (mm)	Shop Name	Mass of Available Size (kg)	Cost/kg	Cost (PKR)	Gas Cutting Charges (PKR)	Total Cost (PKR)
1	Container Cap (Bottom)	200 x 200	203.2 x 203.2	30	29.21	Iron Steel Works	9.47	90 rupees/kg	852	150	1,002
Mild Steel was donated by PNSC Workshop. Therefore, we did not purchase it. However, the rates were found to be as shown above.											

Table C.4 – Stock Dimensions of Material (3)

Acrylic - Only Available In Flat Sheets (Urdu Bazaar, Qasim Centre, Opposite Aurangzeb Park, Karachi)

S. No	Part Name	Sheet Quantity	Required Area For Machining (ft)	Available Area (ft)	Required Thickness For Machining (mm)	Available Thickness (mm)	Shop Name	Cost / Sq. ft	Cost (PKR)	Cutting and Extra Charges	Total Cost (PKR)
1	Middle Disc and Bottom Disc	1	1.25 x 1.25	1.25 x 1.25	5	5.5	Acrylic Centre	300 rupees/Sq. ft	468.75	332.5	970
2	Top Most Plate	1	0.75 x 0.7	0.75 x 0.75				168.75			
3	Canopy Roof	1	0.39 x 0.39	0.39 x 0.39				400 rupees/Sq. ft	60.84	None	168
4	Canopy Boundary Wall	1	0.22 x 1.22	0.22 x 1.22					107.36		

Canopy (Roof + Boundary) is made by Acrylic Technician at Urdu Bazaar. The flat acrylic sheet was heated using laser or flame for making circular canopy. The material for canopy was purchased by the technician himself. However, the rates were found to be as shown above.

Table C.5 – Stock Dimensions of Material (4)

Ertalon (Gear Plastic/Teflon) - Available In Flat Sheets (Shahra-e-Liaquat, Sindh Madrasa, Karachi)									
S. No	Part Name	Required Area For Machining (mm)	Available Area (mm)	Required Thickness For Machining (mm)	Available Thickness (mm)	Shop Name	Mass of Available Size (kg)	Cost/kg	Total Cost (PKR)
1	Top Endcap	270 x 270	270 x 270	42	42	Gulsons Corporation	3.524	1100 rupees/kg	3870

Table C.6 – Stock Dimensions of Material (5)

Aluminum 6061 - Available in the form of Tube/Pipe/Cylinder (Nanakwara, Paan Mandi, Karachi)										
S. No	Required Outer Dia for Machining (mm)	Available Outer Dia for Machining (mm)	Required Inner Dia for Machining (mm)	Available Inner Dia for Machining (mm)	Length of Tube (mm)	Tube Thickness (mm)	Shop Name	Mass of Tube (kg)	Cost/kg	Total Cost (PKR)
1	160	160	140	140	570	10	Quality Metals	7.25	800 rupees/kg	5800

Table C.7 – Stock Dimensions of Material (6)

Appendix D

Simulations

Stress Analysis on Shell:

Stress analysis was performed on the outer shell to determine the maximum deformation when the loads are applied for different thicknesses. The minimum factor of safety obtained through three different failure criteria is above 2. So, 5mm is chosen as the thickness of the shell except for the top and bottom. The study is shown below:

Loads and Fixtures:

The top and bottom part of the Shell were chosen as the effective fixtures during the study. Loads were applied on the circular surfaces on the shell. An inside pressure of 250 psi was applied. Atmospheric pressure (101325 pascals) was applied on the outer circular surface.

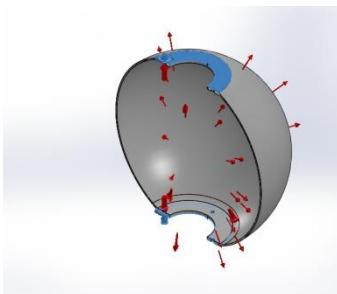
Fixture name	Fixture Image	Fixture Details
Fixed-1		Entities: 2 face(s) Type: Fixed Geometry

Table D.1 – Stress Analysis Study (Fixture)

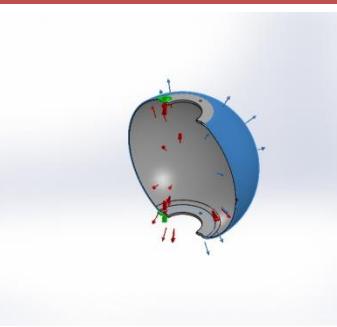
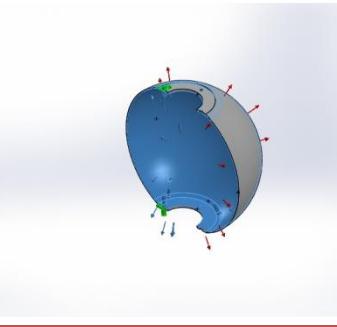
Load name	Load Image	Load Details
Pressure-1		Entities: 1 face(s) Type: Normal to selected face Value: 101325 Units: N/m ²
Pressure-2		Entities: 3 face(s) Type: Normal to selected face Value: 250 Units: Psi

Table D.2 – Stress Analysis Study (Applied Load)

Study Results

As the fiber glass exhibits somehow between ductile and brittle behavior, hence, the factor of safety is calculated using the following criterion:

- Distortion Energy theory (Maximum Von Mises)
- Maximum Normal Stress
- Maximum Shear Stress

The results obtained are summarized below:

Criterion	Maximum Stress (MPa)	F.O.S
Distortion Energy Theory (Max Von Mises Stress)	61.7	3.35
Maximum Normal Stress Theory (Max Principal Stress)	77.2	2.7
Maximum Shear Stress Theory (Max Shear Stress)	34.8	2.97

Table D.3 – Results Obtained Using Different Criterion

Maximum Normal Stress Theory (MNS) proves to be slightly conservative than the other two theories. The maximum Von Mises stress (stress concentration) developed in the shell due to the application of the loads is 61.7MPa.

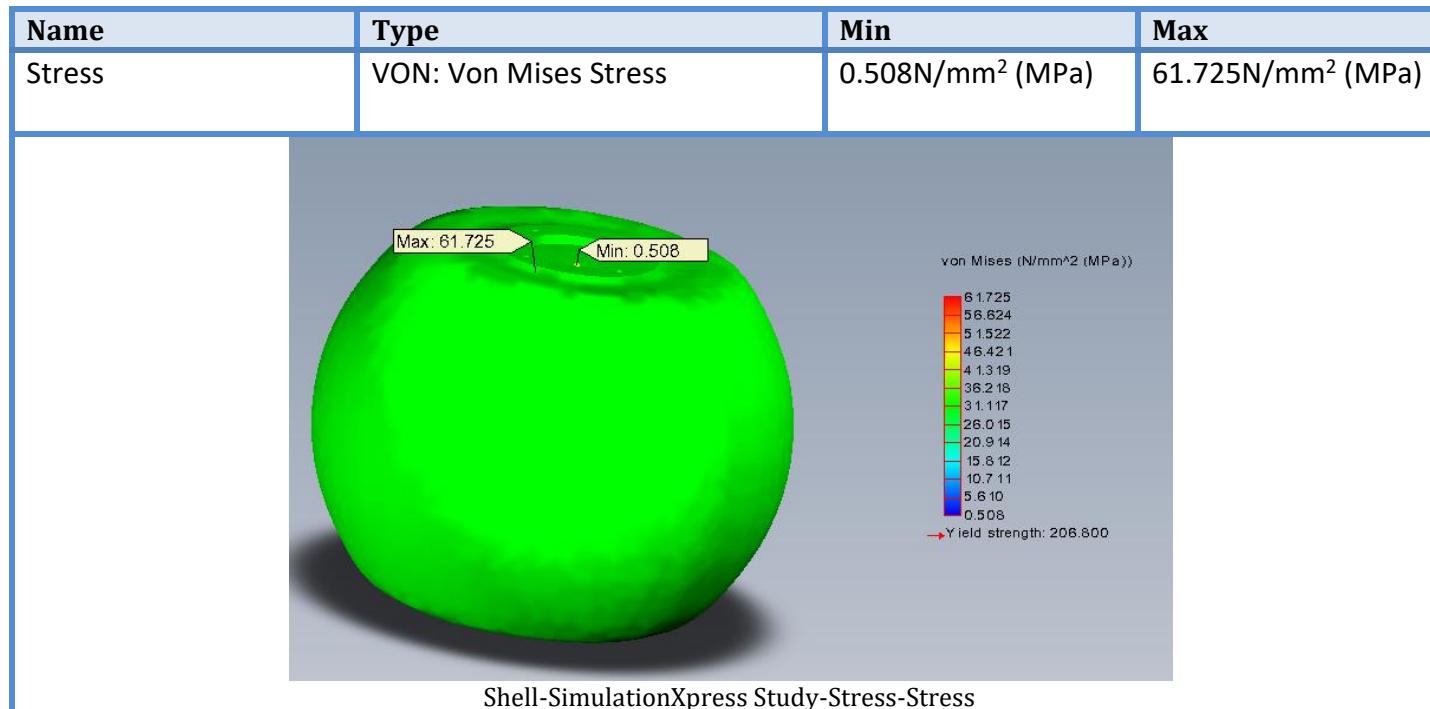


Table D.4 – Stress Analysis Result (Von Mises Stress)

Using the Von Mises criterion, factor of safety is calculated using:

$$F.O.S = \frac{Yield\ Strength}{Maximum\ Von\ Mises\ Stress} = \frac{206.8}{61.725} = 3.35$$

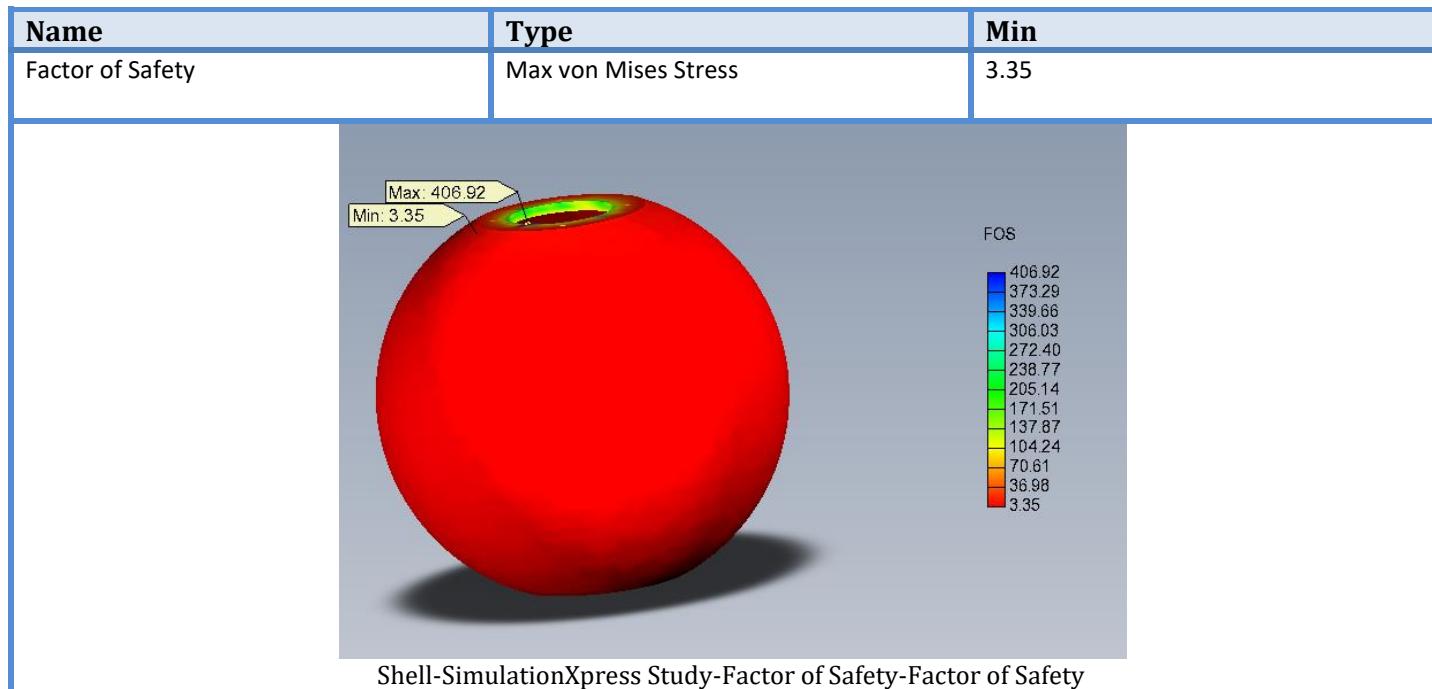


Table D.5 – Stress Analysis Result (F.O.S – Distortion Energy)

According to the Maximum Normal Stress theory, failure occurs if the maximum principal stress (first principal stress) exceeds the yield strength of the material.

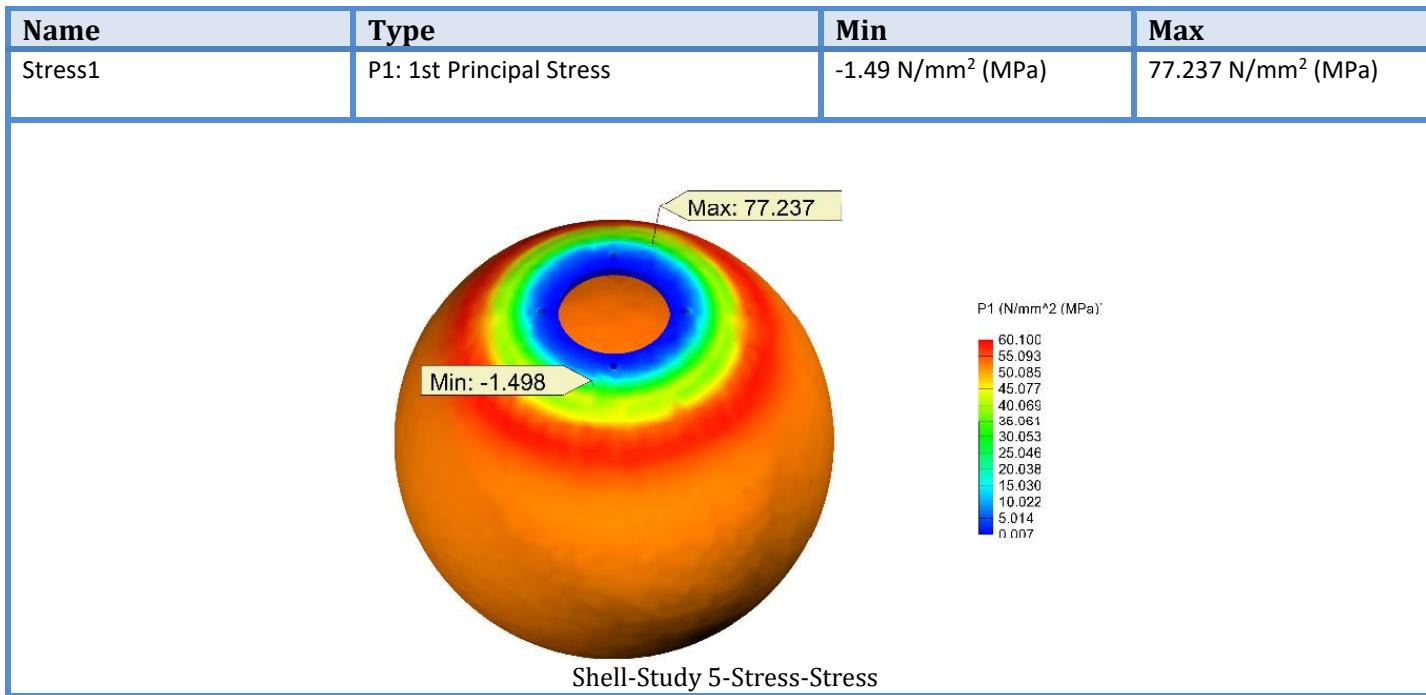


Table D.6 - Stress Analysis Result (Max Normal Stress)

Using the Maximum Normal Stress criterion, factor of safety is calculated using:

$$F.O.S = \frac{Yield\ Strength}{First\ Principal\ Stress} = \frac{206.8}{77.2} = 2.7$$

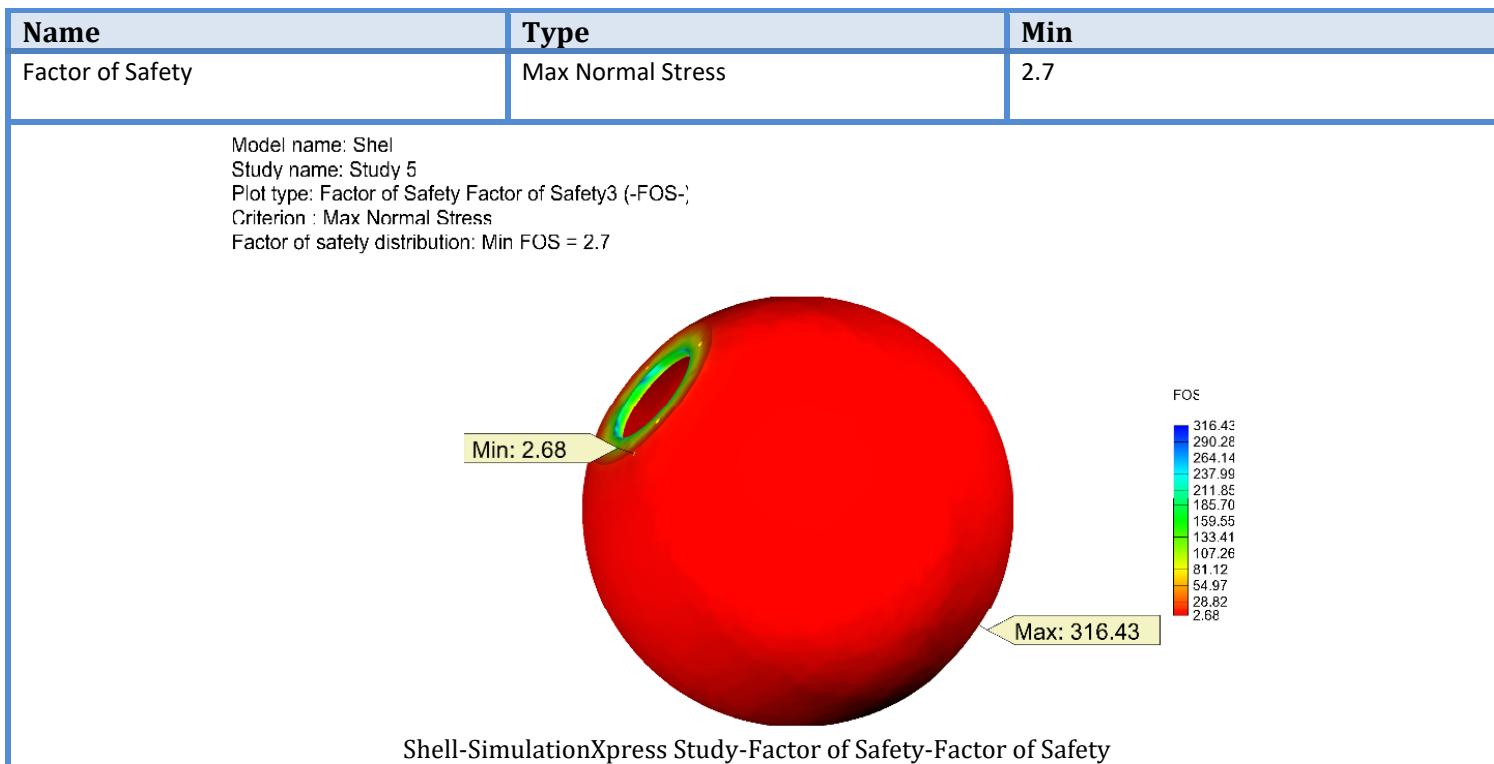


Table D.7 – Stress Analysis Result (F.O.S – Max Normal Stress)

According to Maximum shear stress theory, failure occurs if maximum shear stress equals or exceeds the shear stress at yielding. The shear stress at yielding is half of the yield strength of any material.

$$\text{Shear Strength at Yielding} = \frac{\text{Yield Strength}}{2} = \frac{206.8}{2} = 103.4 \text{ MPa}$$

$$\text{Maximum Shear Stress} = \frac{\text{First Principal} - \text{Third Principal}}{2} = \frac{P_1 - P_3}{2} = \frac{69.6}{2} = 34.8 \text{ MPa}$$

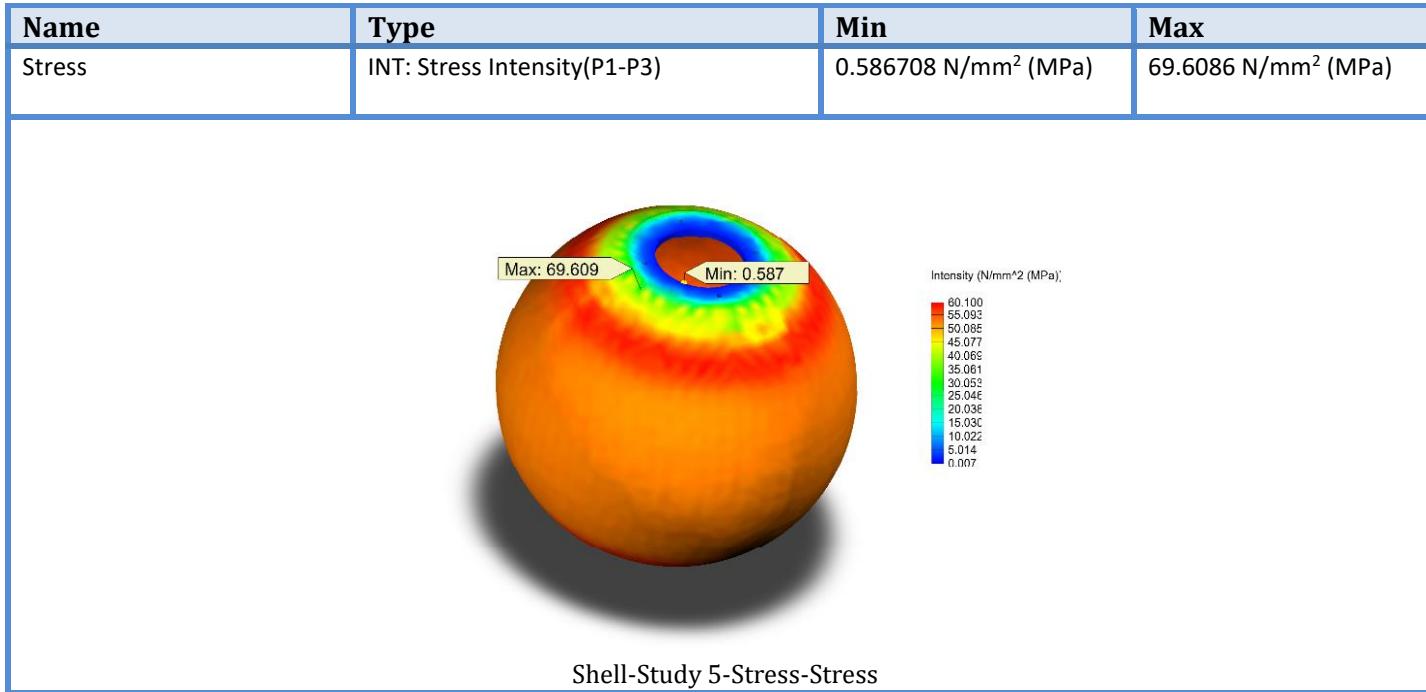


Table D.8 – Stress Analysis Result (Maximum Shear Stress)

Using the Maximum shear stress criterion, factor of safety is calculated using:

$$F.O.S = \frac{\text{Shear strength at yielding}}{\text{Maximum Shear Stress}} = \frac{103.4}{34.8} = 2.97$$

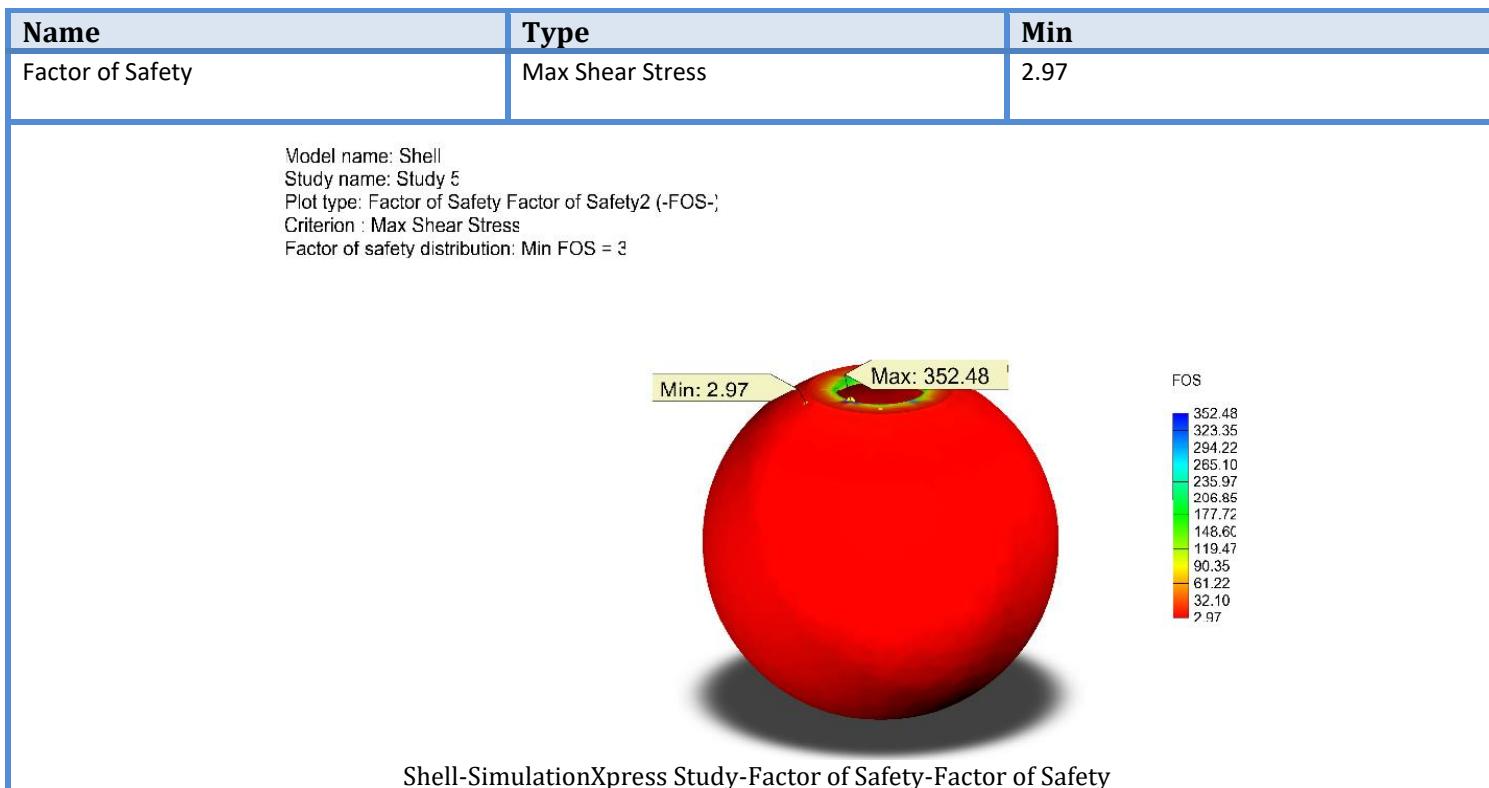


Table D.9 – Stress Analysis Result (F.O.S - Max Shear Stress)

The maximum deformation of 0.19 mm occurs at the sharp edges inside the shell (see Table D.10). Although, 0.19 mm is considered to be in the safe range, the fiberglass shell is manufactured such that the sharp edges are blended to avoid stress concentration as a precautionary measure.

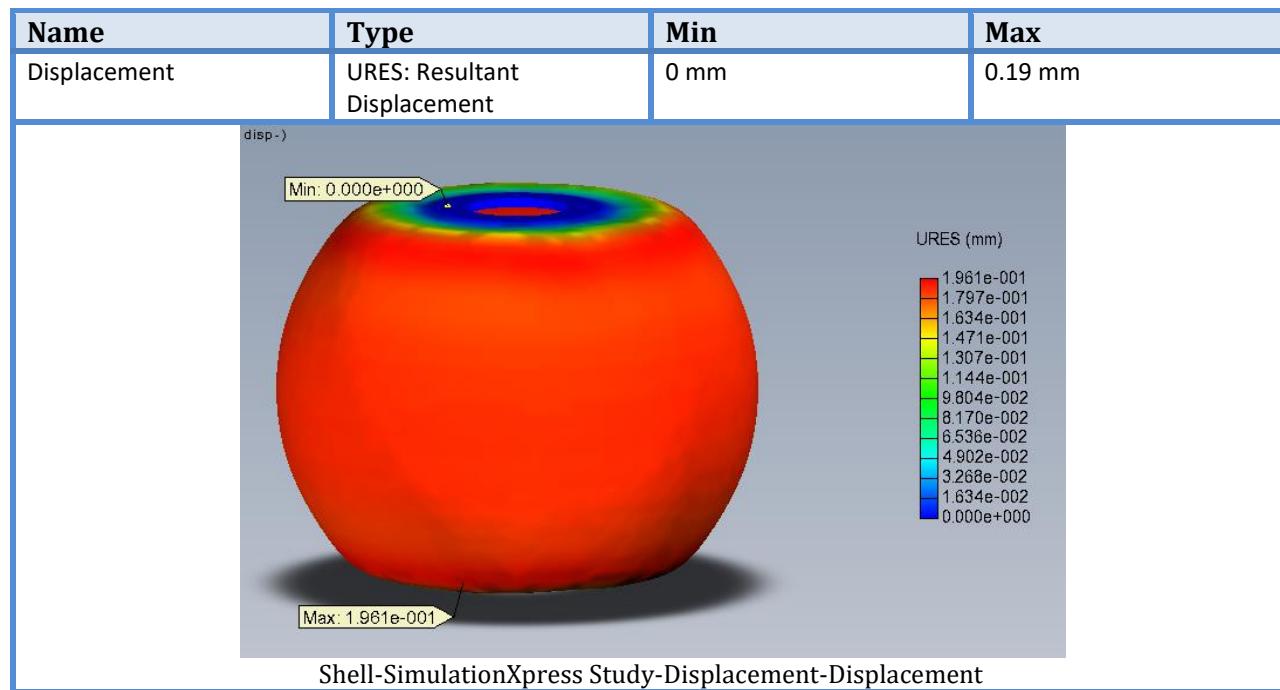


Table D.10 - Stress Analysis Result (Displacement)

Top and Bottom Shell Thickness Optimization Calculation:

A similar calculation was performed to determine the top and bottom thickness of the shell using stress analysis in SolidWorks. The deflection of the top and bottom part should be minimal (approximately 0.08mm) otherwise appropriate compression would not be provided to the O-ring on the metal ring which sits on the top and bottom part of the shell. Atmospheric pressure is applied on the outer circular surface. Results are calculated for internal pressures of 150 and 250 psi. The maximum deflection for 15 mm thickness and 250 psi was found to be 0.06mm with a factor of safety of 1.33. Hence 15mm was chosen as the thickness of the top and bottom part.

Thickness(mm)	Applied Internal Pressure – Pin (psi)	Deflection(mm)	F.O.S (minimum)
5	150	0.19	0.42
5	250	0.3	0.26
10	150	0.1	0.8
10	250	0.18	0.44
15	150	0.03	2.66
15	250	0.06	1.33

Table D.11 – Stress Analysis (Top and Bottom Shell Thickness)

Appendix E

Fasteners

Socket Head Caps are used in most of the assembly due to their strength and design. Taper Head Screws are used to hold container with the caps as they require less space and they are to be flushed with the container.

Part Name	Screw Size (Diameter of head X Length)	Quantity	Screw Series	Name	Nuts (I.D)	Washers (I.D)
Ring to Shell	3/8 X 2	8	Inch	Socket head caps	Yes (3/8)	Yes (3/8)
Top Endcap to Ring	3/8 X 0.5	4	Inch	Socket head caps	No	Yes (3/8)
Bottom Endcap to Ring	3/8 X 0.5	4	Inch	Socket head caps	No	Yes (3/8)
Top Cover to Top Endcap	3/8 X 1.5	4	Inch	Socket head caps	No	Yes (3/8)
Bottom Cover to Bottom Endcap	3/8 X 1.25 - 3/8 X 1.5	4	Inch	Socket head caps	No	Yes (3/8)
Top Most Plate to Top Cover	M6 X 85 - M6 X 87	4	Metric (mm)	Socket head caps	No	Yes (6)
Top Most Plate to Canopy	M6 X 8 - M6 X 10	4	Metric (mm)	Socket head caps	No	Yes (6)
Container to Container Caps	M6 X 7.5 - M6 X 8	8	Metric (mm)	Socket button	No	Yes (6)
Threaded Rods	M10 X710	42	Inch	--	Yes (3/8)	Yes (3/8)

Table E.1 – Fasteners Used

Appendix F

Machine Drawings

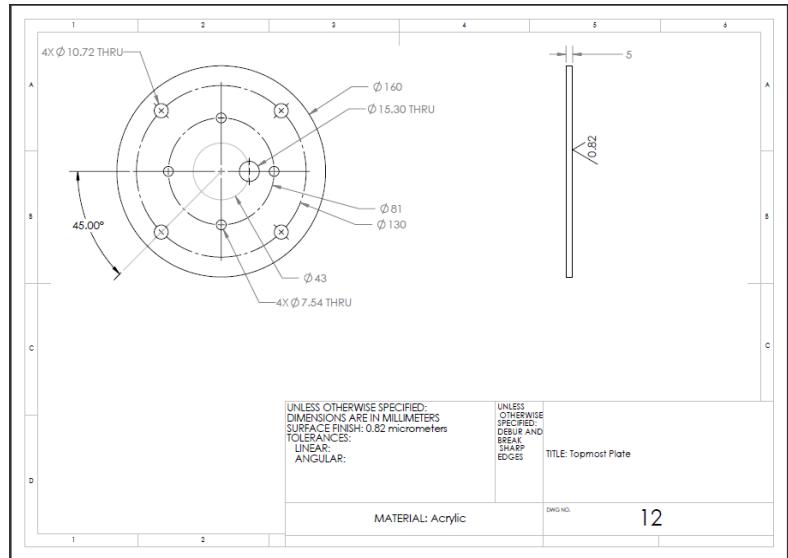


Figure F-1 – Top Most Canopy Plate

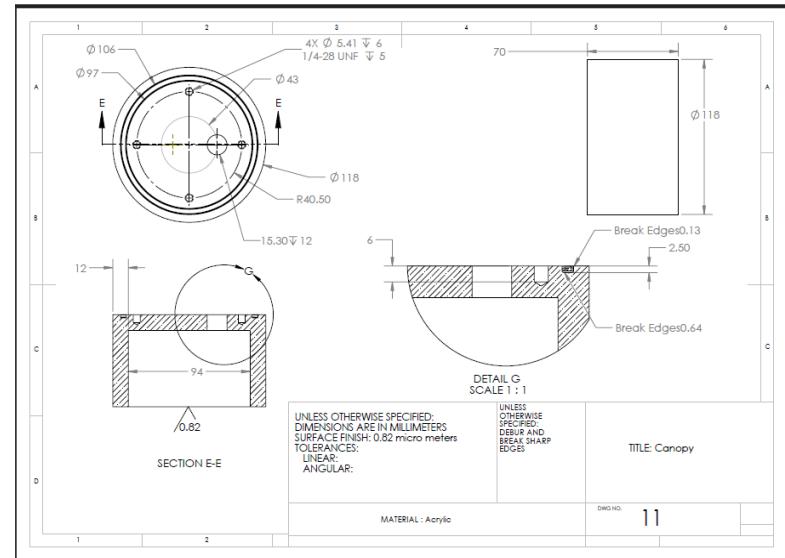


Figure F-2 – Canopy

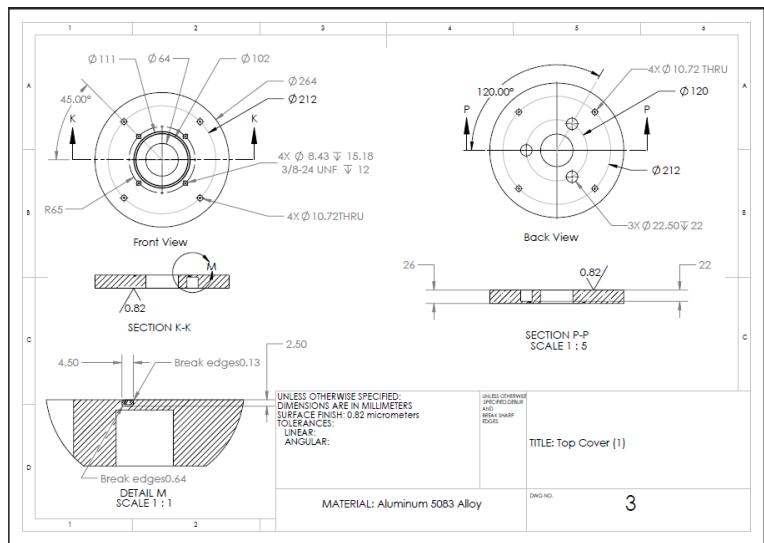


Figure F-3 – Top Cover

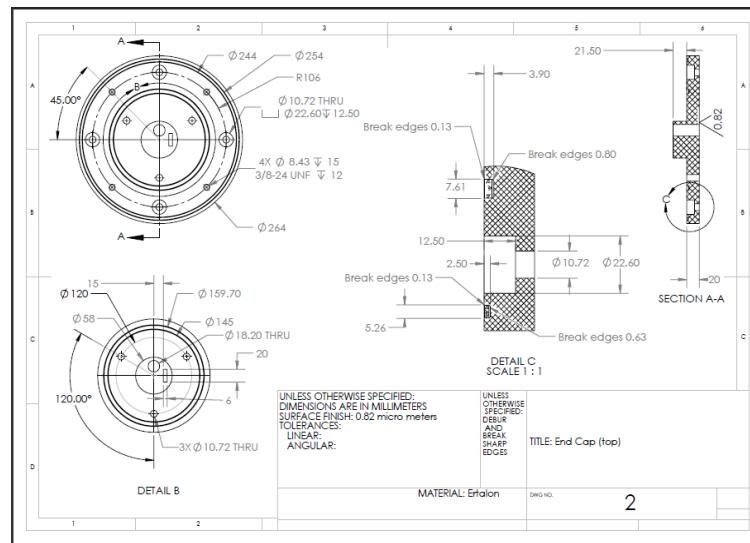


Figure F-4 – Top Endcap

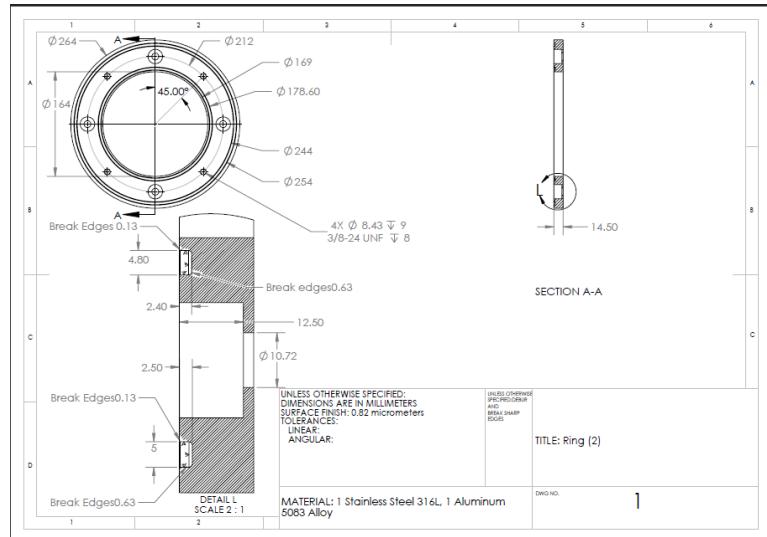


Figure F-5 – Top and Bottom Ring

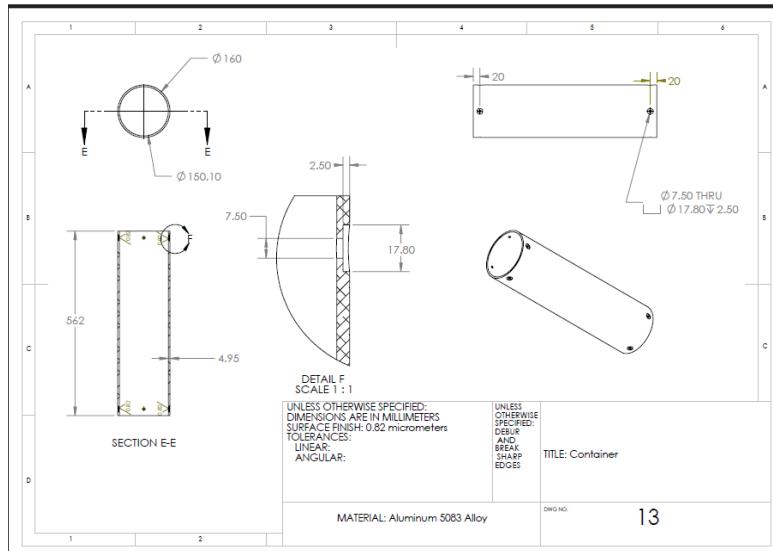


Figure F-6 – Container Boundary

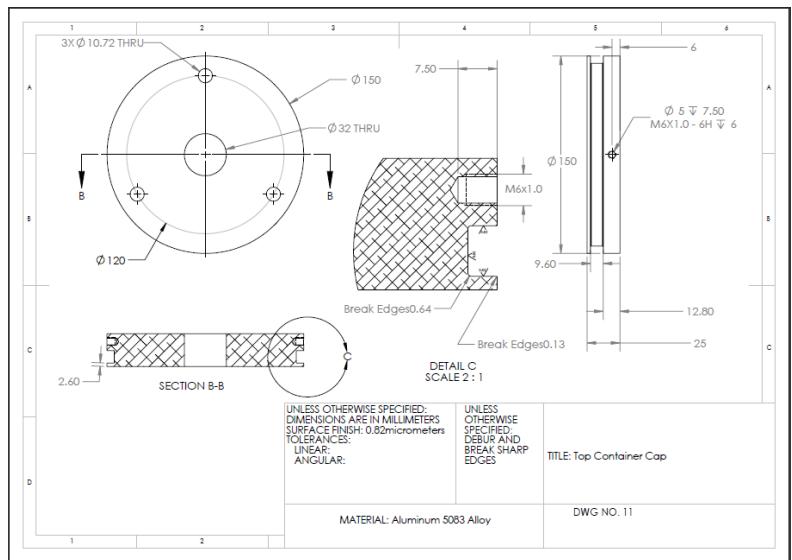


Figure F-7 – Top Container Cap

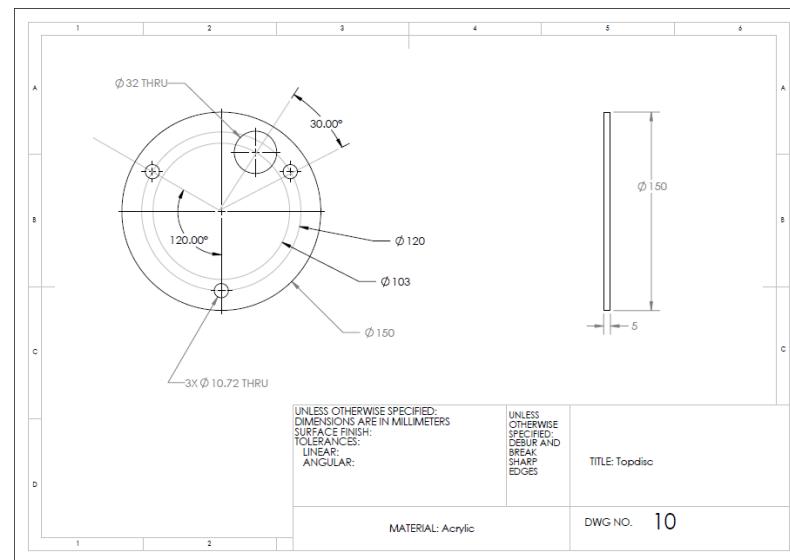


Figure F-8 – Top Disc

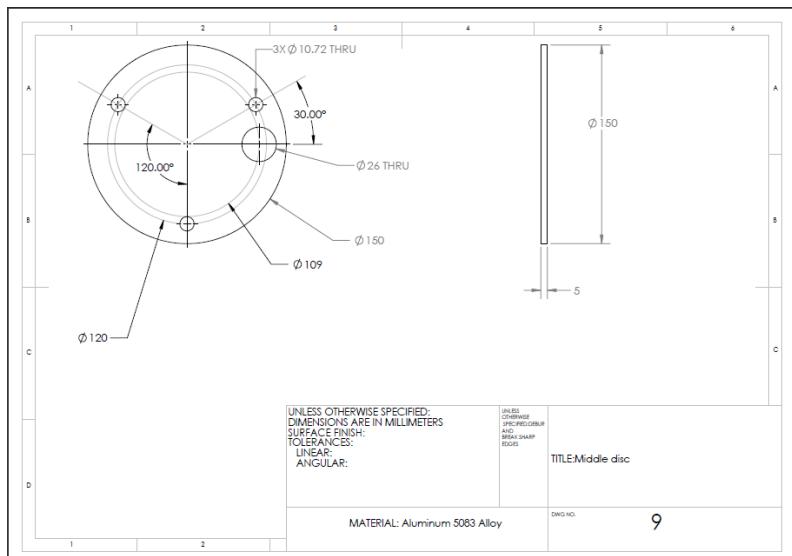


Figure F-9 – Middle Disc

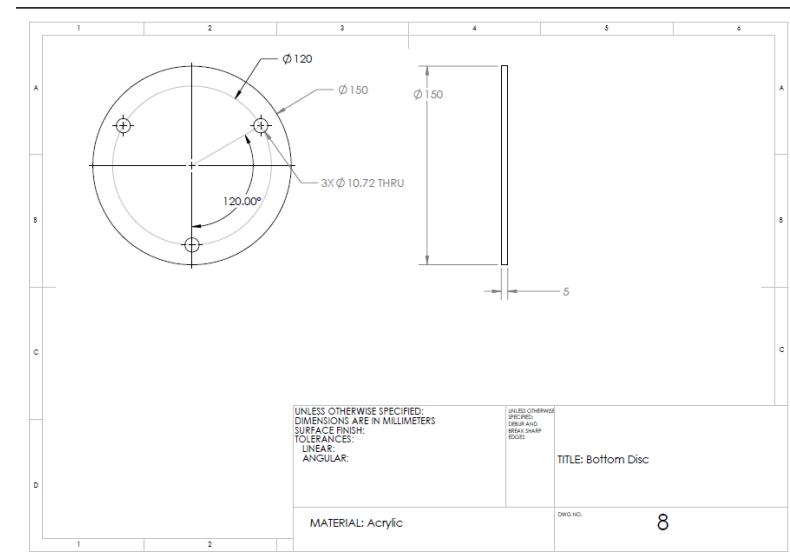


Figure F-10 – Bottom Disc

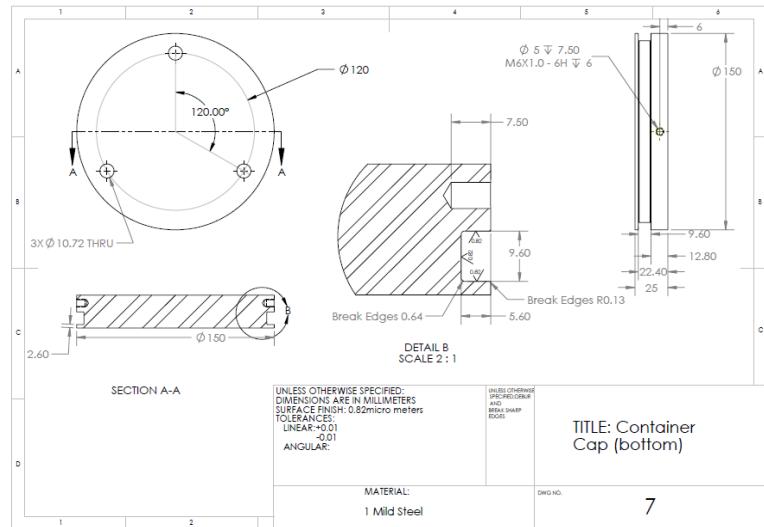


Figure F-11 – Bottom Container Cap

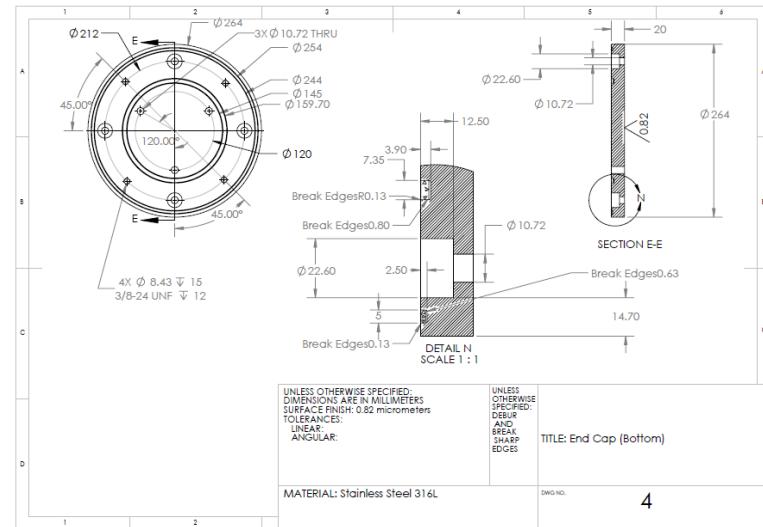


Figure F-12 – Bottom Endcap

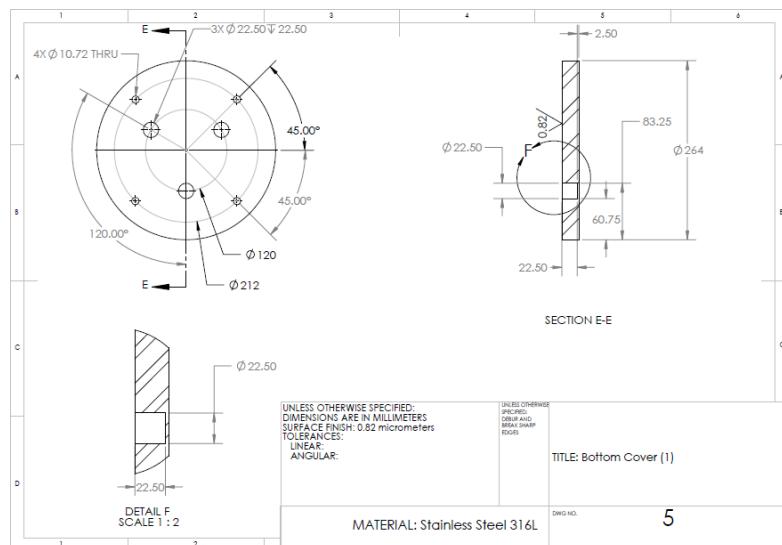


Figure F-13 – Bottom Cover

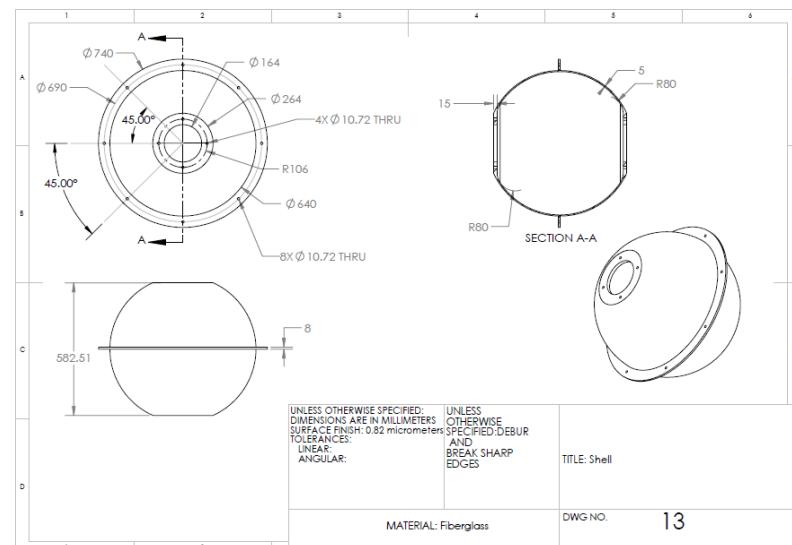


Figure F-14 – Fiberglass Shell

Appendix G

Mechanical Seals/O-rings

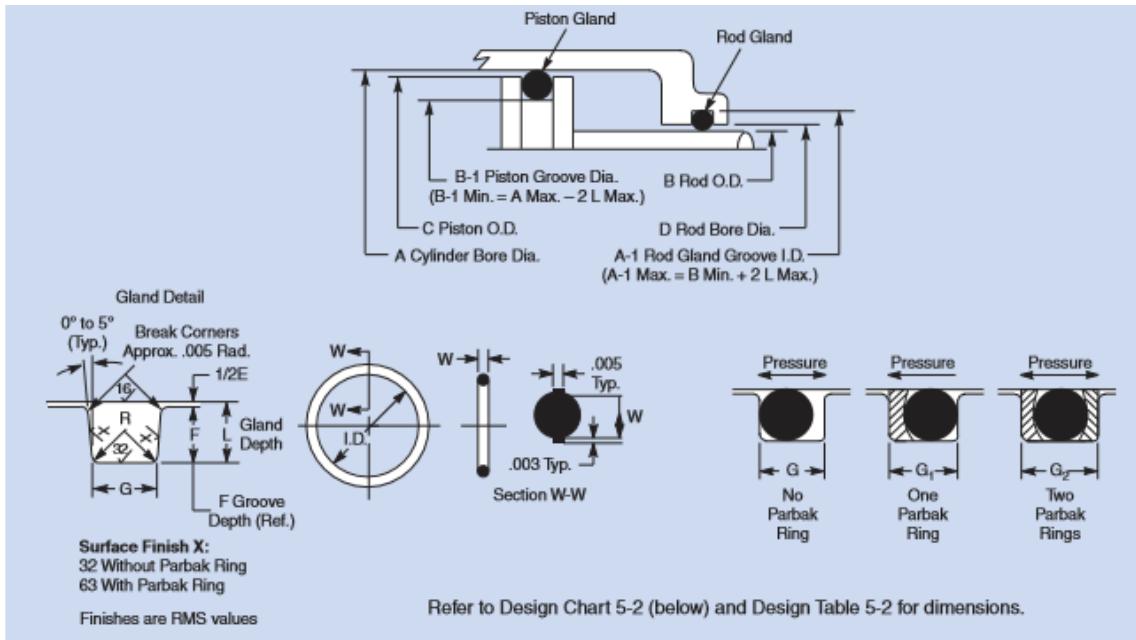
1. O-ring Sizes Used

Face Seal O-ring										
S. No	Part Name	Seal Name	Material	Quantity	Required			Available		
					Outer Dia	Inner Dia	Thickness	Outer Dia	Inner Dia	Price (PKR)
					(mm)	(mm)	(mm)	(mm)	(mm)	
1	Top Ring	Inner Seal	Buna-N (NBR)	1	177.5	170.5	3.5	176	170	3
		Outer Seal		1	253	246	3.5	252	245	300
2	Bottom Ring	Inner Seal		1	177.5	170.5	3.5	176	170	300
		Outer Seal		1	253	246	3.5	252	245	300
3	Top Endcap	Inner Seal	Buna-N (NBR)	1	158.1	147.5	5.3	158	147	5.5
		Outer Seal		1	253	246	3.5	252	245	300
4	Bottom Endcap	Inner Seal		1	158.1	147.5	5.3	158	147	5.5
		Outer Seal		1	253	246	3.5	252	245	300
5	Canopy	Canopy Seal	Buna-N (NBR)	1	105	98	3.5	105	98	3.5
6	Top Cover	Top Cover Seal		1	110	103	3.5	110	103	300
Dynamic Seal O-ring										
7	Container Cap	Top Cap Seal	Buna-N (NBR)	1	154	140.2	6.9	154	140	7
		Bottom Cap Seal		1	154	140.2	6.9	154	140	7

Table G.1 – O-ring Sizes Used

2. Dynamic O-ring Calculation

Following calculation was performed to provide the O-ring sufficient compression to work properly:



O-Ring Size	Dimensions			A	A-1	B	B-1	C	D	G†	
Parker No.				Mean	Bore Dia. (Cylinder)	Groove Dia. (Rod Gland)	OD (Rod)	Groove Dia. *(Piston)	OD (Piston)	Bore Dia. (Rod)	Width Groove
	I.D.	\pm	W	O.D. (Ref)	.+002 -.000	.+.004 -.000	.+.000 -.002	.+.000 -.004	.+.000 -.001	.+.001 -.000	.+.005 -.000
434	142.24	0.93	6.98	156.2	155.6	154.8	142.8	143.58	155.5	142.9	9.52

Figure G-1 – O-ring Squeeze Calculation [14]

For Calculating the Compression on the O-ring:

$$\text{Compression} = A - C = 155.6 \text{ mm} - 155.5 \text{ mm} = 0.1$$

Appendix H

Project Expenditure

1. Stock Material Cost

Material Name	Material Cost (PKR)	Material Name	Material Cost (PKR)	Total Material Cost (PKR)
SS - 316L	23,400	Ertalon	3,870	40,040
Al - 5083	6,000	Acrylic	970	
Mild Steel	Donated	Al - 6061	5800	

Table H.1 – Stock Material Cost

2. Cost of Electronic Components

Component	Description	Quantity	Spent (USD)
13 dBi Antennas	For XBee - 13 dBi Implies larger power transmission but narrow signal broadcast	2	12.22
7 dBi Antennas	For XBee - 7 dBi Implies slightly lower Transmission but goes in a broader direction	3	9.54
Waterproof Plastic Boxes	For instruments storage	2	11.06
Step Down DC-DC Buck	6-24 V stepped down to 5 V for LiPo battery as most electronics need ~5V	2	7.02
XBee Shield for Remote PC	XBee Shield for Slave Arduino	1	6.56
DHT11 Sensor	Humidity + Temperature inside buoy if needed	1	2.49
Zip Ties / Velcro	Misc. Arduino cable organization equipment	-	5.85
Ribbon Cables + Connector	For cleaner look of wires	-	4.15
XBees	RF Module to Transmit Data Wirelessly over long range	2	89.9
XBee Antenna Cables	Cable to Attach XBee to XBee Antenna	2	10.56
IMU 1 - BNO 055	9 DoF, 16-bit Accelerometer, 16-bit gyrometer	1	34.96
IMU 2 - LSM 9DS0	9 DoF, 14-bit Accelerometer, 16-bit gyrometer	1	24.95
Raspberry-Pi	Processor for handling camera	1	29.99
Raspberry-Pi camera	The camera built for the Raspberry Pi	1	29.99
Raspberry-Pi camera case	To fit the camera inside the canopy	1	6.95
Raspberry-Pi Heatsink	To avoid the Raspberry-pi from overheating	1	1.95
Arduino Mega 2560	The main processor	1	9.99
Arduino Wi-Fi Shield	The Board for Holding the XBee and SD Card Logger	1	39.99
LiPo Battery*	7.4 V, 1000 mAh battery to power Electronics	1	0
GPS Module and Antenna*	To locate XBee in a drifting mode	1	0
Flashing LED*	Act as a blinking indicator for clear visibility of buoy at night	1	0
Adafruit Shipping	Adafruit Shipping Charges		8.59
DHL Shipping	DHL Shipping Charges		166.04
* Donated			Total Cost
			USD 512.75
			PKR 53,644

Table H.2 – Cost of Electronic Components

3. Total Project Expenses

Total Project Expenses			
S. No	Budget Items	Cost (PKR)	Total Cost (PKR)
1	Stock Material	40,040	
2	Electronics	53,644	
3	Fiberglass Shell	50,000	
4	Threaded Rods	3,640	
5	Mechanical Seals/O-rings	3,600	
6	Fasteners (Bolts, Nuts and Washers)	4,536	
7	Dow Corning 1000 Anti-Seize Paste	1,200	
8	Molykote 55 O-ring Silicone Grease	1,250	
9	Epoxy, Cable/Zip Ties, Duct Tape, Double Tape, Alcohol Swabs	1,150	
10	Latex Gloves	160	
11	Aluminum Tube	5,840	
12	Container Cap O-rings	400	
13	Taper Head SS Bolts and Washers	120	
14	Transport/Commuting Charges (Careem-Service, Suzuki, Rickshaw)	8,085	
15	Water to Fill Water Pond For Testing (Water Tanker Charges)*	5,000	
Wasted Items			
15	HDPE (Top Container Cap and Endcap)	2,367	
16	O-rings (Wrong Size)	1,000	
17	Canopy Bolts and Washers (Wrong Size)	250	
18	Acrylic Container and its fabrication (Low Precision Machining)	3,500	
* Donated			

Table H.3 – Total Project Expenses

Appendix I

Electronics Layout

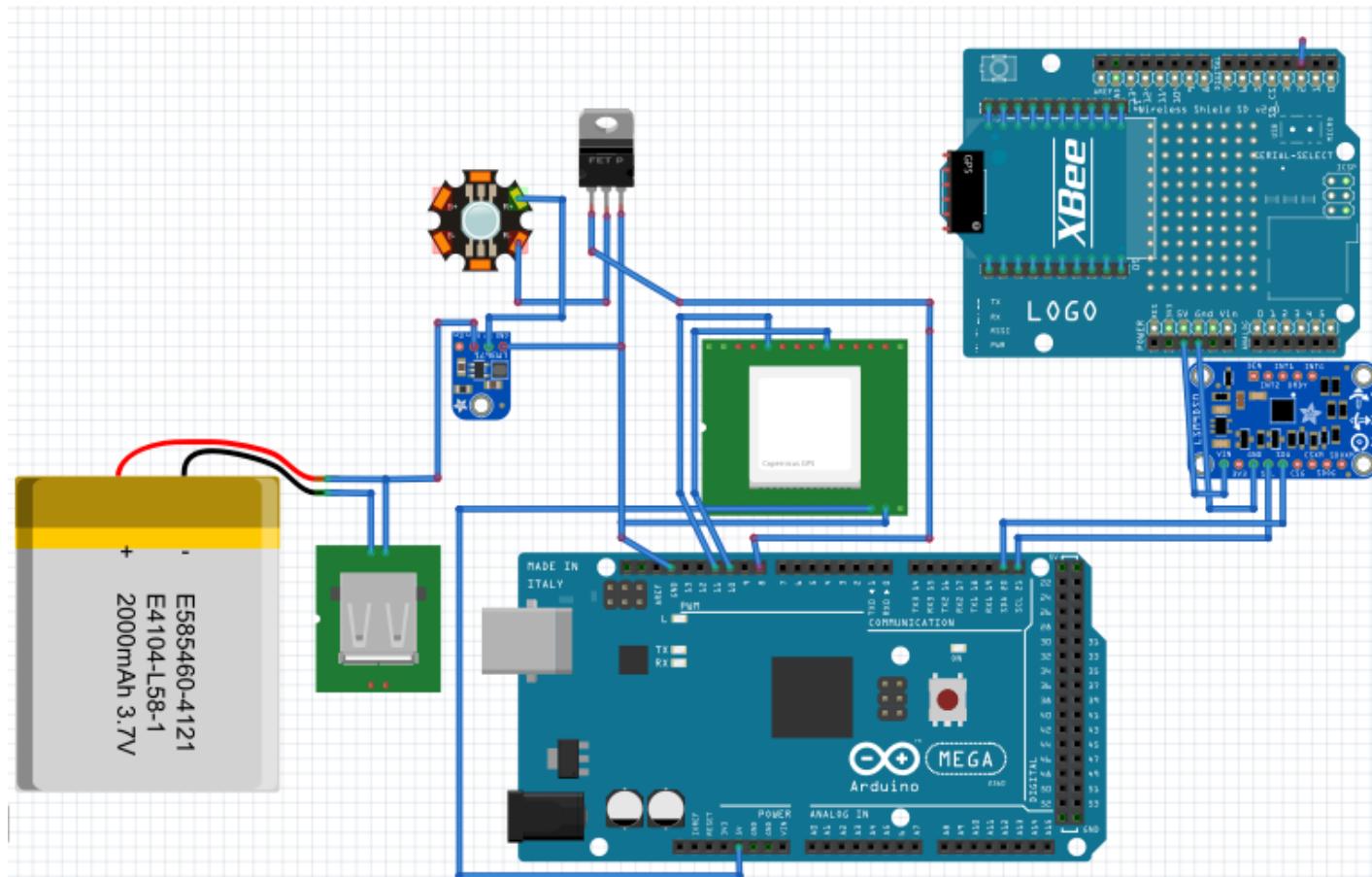


Figure I-1 – Fritzing Electronics Schematic

Appendix J

Hardware

1. List of Major Electronic Components Used
 - i. Arduino Mega 2560



Figure J-1 – Arduino Mega 2560

An Arduino Mega 2560 was used for this project because of its simplicity and standard usage in university at the undergraduate level. The Mega is superior to the Uno which has a larger number of pins allowing for more sensors to be connected and a larger program memory which was required for this project, at the cost of a slightly larger size.

ii. LSM9DS0 Accelerometer Unit



Figure J-2 – LSM9DS0

The accelerometer and gyroscope with 9 DoF is used to determine the orientation of the waves at any given instant. There were plenty of IMUs to choose from, and the LSM9DS0 was chosen for its low cost, high resolution and greater linear acceleration sensitivity specifications and its ease of use with fast mode I2C communication and existing libraries.

iii. Trimble GPS Module & Antenna



Figure J-3 – Trimble GPS & Antenna

A GPS module has been used to log the location of the data buoy at set intervals. Trimble was selected due to its very good quality for the price. An antenna protruding towards the top cover has been used to enhance the signals.

iv. LiPo Battery



Figure J-4 – LiPo Battery

The LiPo battery is the primary source of power for all of the electronics sitting in the data buoy. Its selection is dependent on the overall power consumption of the buoy.

v. Buck Converter



Figure J-5 – 5 Volt Buck Converter

This sole purpose of the buck-module is to step-down the voltage provided by the LiPo battery to a stable, 5 V required by the Arduino module.

vi. Flashing LED



Figure J-6 – Flashing LED

Used as visual aid and secondary means to identify the location of the data buoy. Because a high intensity light flash is needed for greater visibility, a high-current LED module has been used.

vii. XBee Radio Module



Figure J-7 – XBee Pro Series 2B Module

The XBee is a minuscule radio link module that can wirelessly transmit data using the Arduino's serial communication. The Pro modules in particular boast a long range (1 mile).

viii. Arduino Wireless SD Shield



Figure J-8 – Arduino Wireless SD shield

The Wireless SD shield accommodates the XBee module and also possesses a slot for the on-board SD card. A switch on the board allows the programmer to switch between XBee programming mode and XBee transmission mode. Furthermore a prototyping soldering board exists.

2. Specifications of Major Electronic Components

i. Arduino Mega 2560

Operating Voltage	5	V
Digital I/O Pins	54	
Analog Input Pins	16	
DC Current per I/O Pin	20	mA
DC Current for 3.3V pin	50	mA
Flash Memory	256	KB
SRAM	8	KB
EEPROM	4	KB
Clock Speed	16	MHz
Weight	37	g

Table J.1 – Arduino Mega 2560 Specifications

ii. LiPo Battery

Battery Capacity	1000	mAh
Voltage	7.4	V
Cell Configuration	2S1P	
Discharge Rate	15	C

Table J.2 – LiPo Battery Specifications

iii. **Copernicus II Trimble GPS**

Horizontal Accuracy	< 2.5	m
Altitude Accuracy	< 2.5	m
Hot Start	3	s
Warm Start	35	s
Cold Start	38	s
Input Voltage	2.7 – 3.3	V
Current Consumption	44	mA
Power Consumption	132	mW
Protocol(s)	NMEA, TSIP, TAIP	

Table J.3 – Copernicus Trimble GPS Specifications

iv. **XBee Pro S2B Module**

Indoor Range	90	m
Outdoor Range	1600	m
Transmit Power Output	63	mW
RF Data Rate	250,000	bps
Supply Voltage	3.3	V
Transmit Current	250	mA
Operating Frequency	2.4	GHz
Operating Temperature	-40 to 85	°C

Table J.4 – XBee Pro S2B Module Specifications

v. **LSM9DS0 Accelerometer Unit**

<i>Major Specifications at V = 3.0 V, T = 25 °C</i>		
Linear acceleration range	±2	g
	±4	
	±6	
	±8	
	±16	
Angular rate range	±245	dps
	±500	
	±2000	
Magnetic measurement range	±2	gauss
	±4	
	±8	
	±12	
Linear acceleration sensitivity	2 g	0.061
	4 g	0.122
	8 g	0.183
	8 g	0.244
	16 g	0.732
Angular rate sensitivity	245 dps	8.75
	500 dps	17.50
	2000 dps	70
Magnetic sensitivity	2	0.08
	4	0.16
	8	0.32
	16	0.48
Nominal input voltage	3.3	
Current consumption	6.1	
Output data resolution	16-bit	
Interfaces	I2C, SPI	

Table J.5 – LSM9DS0 Accelerometer Specifications

3. Component Current Consumption

Component	Estimated Max Current (mA)
Arduino Mega 2560	50
Arduino Wireless SD shield	200
XBee Transmitter	220
XBee Antenna Cable	10
XBee Antenna	10
GPS Module (Trimble)	44
GPS Module (Antenna)	9
IMU (LSM9DS0)	6.5
DC buck 5V regulator	1
DC buck variable regulator	5
Flashing LED	200
Total (Amps)	0.77

Table J.6 - Component Current Consumption

Appendix K

Software

1. Software Platforms Used

i. X-CTU

X-CTU is the standard software created by Digi for configuring and testing their RF module products, including the XBee. Before using the XBees for any data transfer, they must be configured so that they can communicate with one other. To set up the hardware, each of the XBees was set up on a development shield in USB mode and programmed with an empty sketch. The first step after loading the software was to set the destination address of each XBee to that of the other one and provide them with the same Pan ID address, a number that determines what network the XBees operate on. Communication only occurs if they share the same Pan ID number.

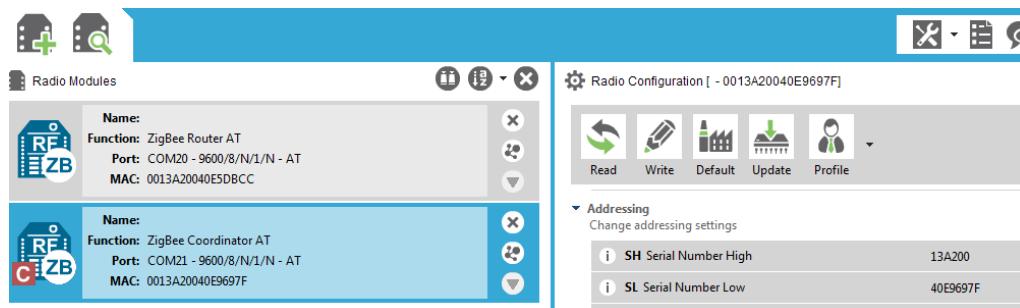


Figure K-1 – XCTU XBee Module Detection

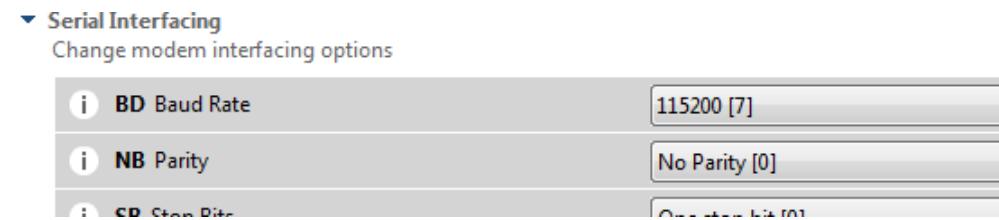


Figure K-2 – XCTU Baud Rate Setting for XBee

ASCII packets were sent from the coordinator XBee in the console log. To verify wireless communication existed, the same data be transmitted to the XBee on the receiving end.

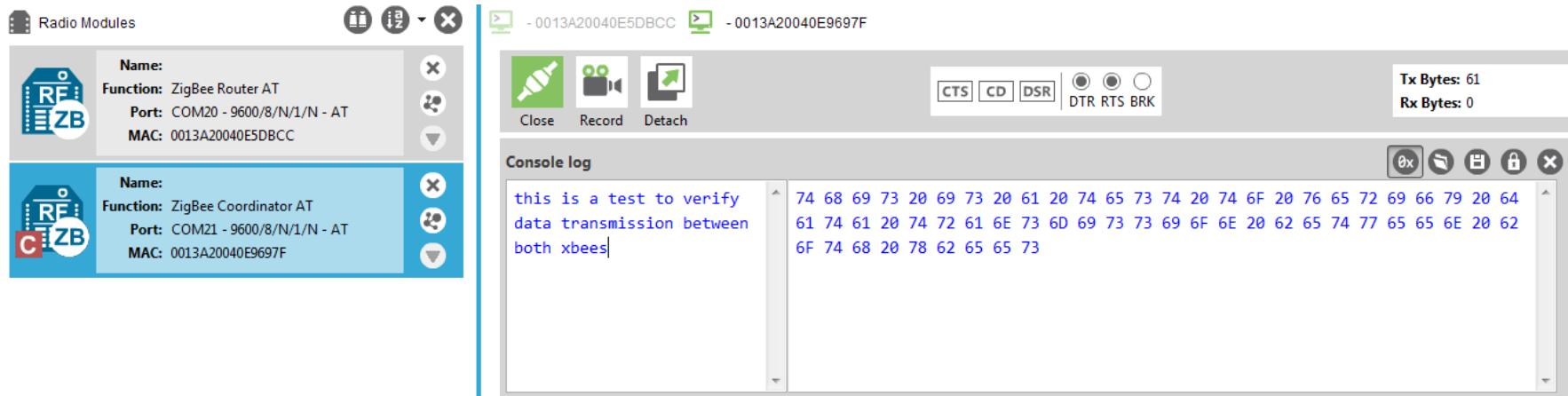


Figure K-3 – XBee Module Data Transmit Test

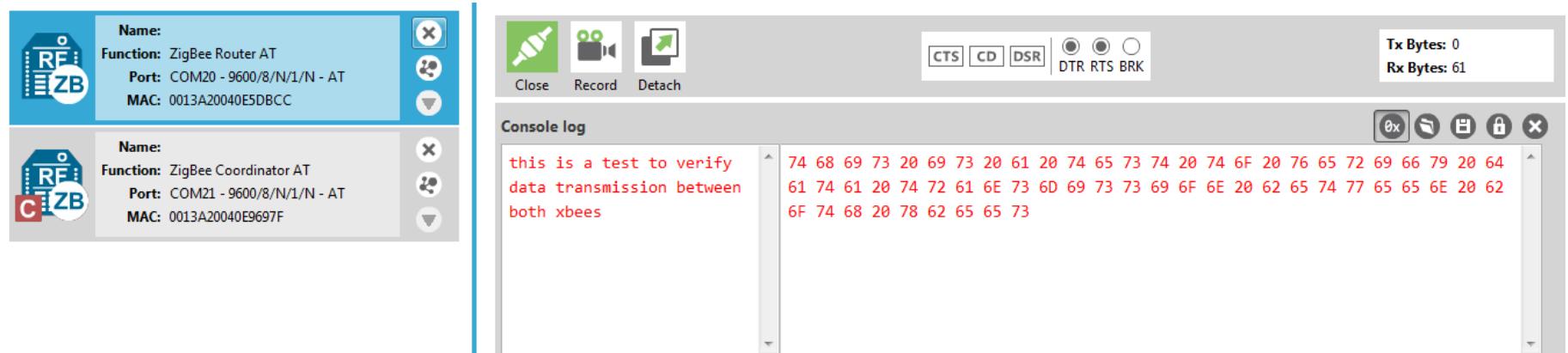
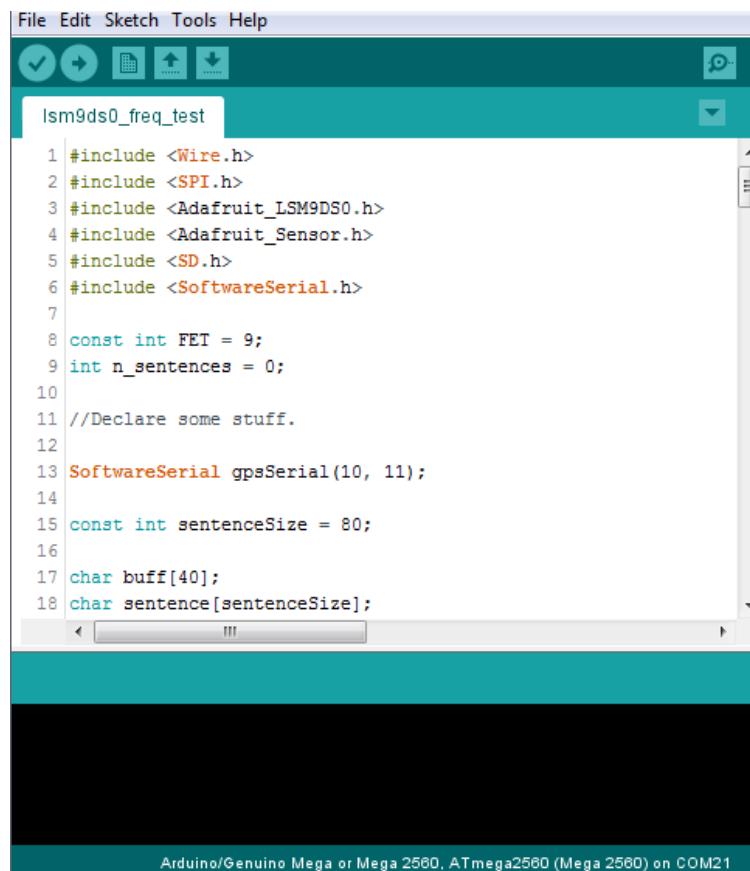


Figure K-4 – XBee Module Data Receive Test

ii. Arduino

Arduino is also the name of the software used to program the Arduino microcontrollers. Code for data acquisition and SD card and serial transmission of the data from the ‘master’ Arduino was written for the Mega 2560, and code to receive the packets of data was programmed for the receiving microcontroller on a Uno. To facilitate coding, preexisting libraries were used for the sensors and I2C communication.



The screenshot shows the Arduino IDE interface. The menu bar includes File, Edit, Sketch, Tools, and Help. Below the menu is a toolbar with icons for saving, loading, and running the sketch. The title bar displays the sketch name: "lsm9ds0_freq_test". The main area contains the C++ code for the sketch:

```
File Edit Sketch Tools Help
lsm9ds0_freq_test
1 #include <Wire.h>
2 #include <SPI.h>
3 #include <Adafruit_LSM9DS0.h>
4 #include <Adafruit_Sensor.h>
5 #include <SD.h>
6 #include <SoftwareSerial.h>
7
8 const int FET = 9;
9 int n_sentences = 0;
10
11 //Declare some stuff.
12
13 SoftwareSerial gpsSerial(10, 11);
14
15 const int sentenceSize = 80;
16
17 char buff[40];
18 char sentence[sentenceSize];
```

At the bottom of the IDE, a status bar indicates: "Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM21".

Figure K-5 – Arduino Software used for Coding the Microcontroller

iii. MS Excel

Once data has been acquired, it needs to be graphed for further analysis. Excel is simple and widely known software that can make some of these plots. MS Excel has the ability to delimit the data and present it in an organized form, which can then be used to create plots. An example of a plot made in Excel is shown below, which is the acceleration data of the LSM9DS0 taken over an 8 second interval.

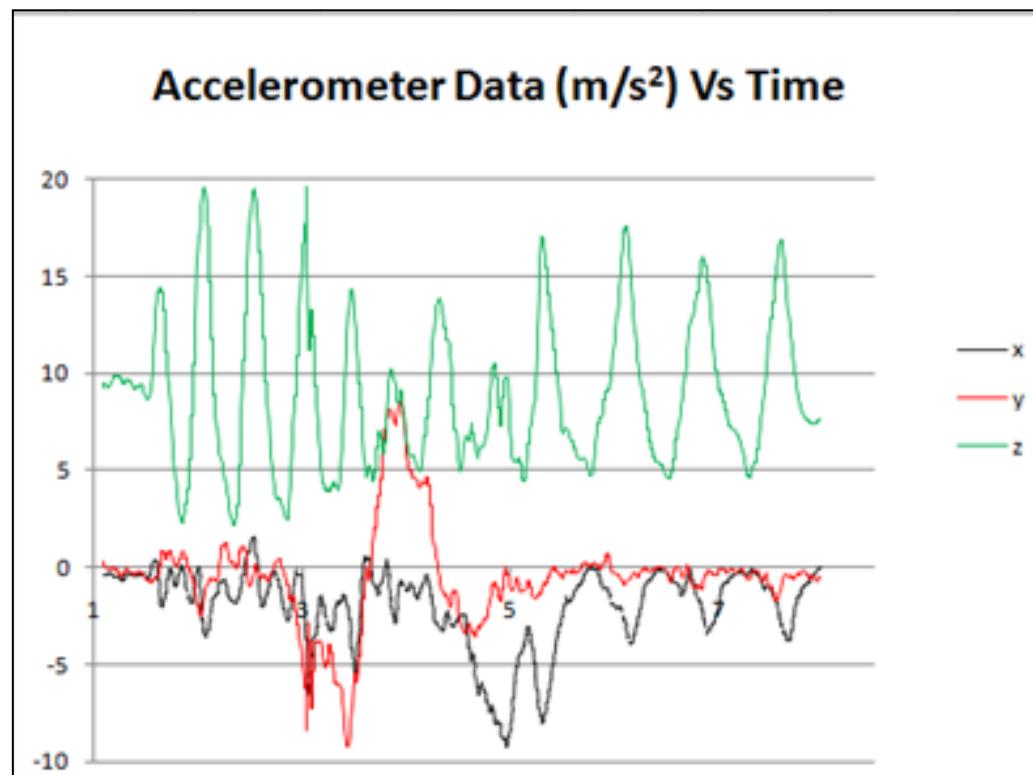


Figure K-6 – Sample Acceleration Data from the Accelerometer

iv. Tracker

Tracker is a free video analysis and modeling tool built on the Open Source Physics (OSP) Java framework. By recording a high-quality video of a contraption that houses an accelerometer, the data from the accelerometer can be corroborated with the analysis done frame-by-frame in the video.

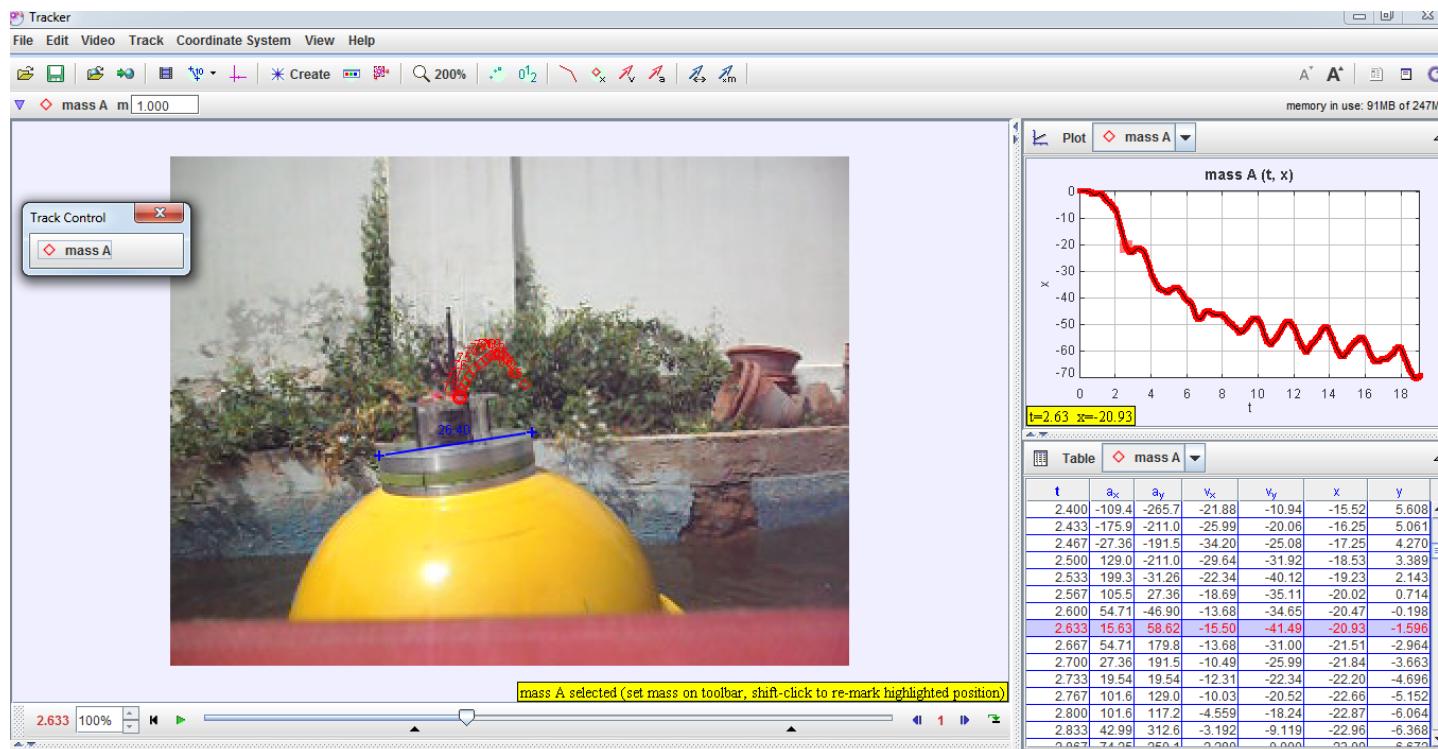


Figure K-7 – Tracker Software Used for Data Reduction

2. Program Code

i. Transmitter

```
#include <avr/sleep.h>

#include <Wire.h>
#include <SPI.h>
#include <Adafruit_LSM9DS0.h>
#include <Adafruit_Sensor.h>
#include <SD.h>
#include <SoftwareSerial.h>

const int FET = 8;
int n_sentences = 0;

SoftwareSerial gpsSerial(10, 11);
const int sentenceSize = 80;
char buff[60];
char sentence[sentenceSize];
String data = "";

File logfile;
Adafruit_LSM9DS0 lsm = Adafruit_LSM9DS0();

//Set sensitivites of accelerometer, gyroscope and magnetometer.
void setupSensor()
{
    lsm.setupAccel(lsm.LSM9DS0_ACCEL RANGE_2G);
    lsm.setupGyro(lsm.LSM9DS0_GYROSCALE_245DPS);
    lsm.setupMag(lsm.LSM9DS0_MAGGAIN_2GAUSS);
}

void setup()
{
    pinMode(FET, OUTPUT);

    gpsSerial.begin(4800);
    Serial.begin(115200);

    Serial.println("Initializing..");
```

```

if (!lsm.begin())
{
    Serial.println("Unable to initialize the LSM9DS0. Check your wiring!");
    while (1);
}
Serial.println("Found LSM9DS0 9DOF.");
Serial.println("");
Serial.println("");

if (!SD.begin(4))
{
    Serial.println("No SD card detected. Data will be transmitted, but not stored."); // No card available?
Let us know.
}
else
{
    Serial.println("SD card detected. Writing contents to Data.txt..");
    logfile = SD.open("Data.txt", FILE_WRITE);
}

delay(200);
displayNMEA();
}

void loop()
{

sensors_event_t accel, mag, gyro, temp;
lsm.getEvent(&accel, &mag, &gyro, &temp);

sprintf(buff, "<%s, %s, %s, %s, %s, %s, %lu>", f2s(accel.acceleration.x, 3), f2s(accel.acceleration.y,
3), f2s(accel.acceleration.z, 3),
f2s(gyro.gyro.x, 3), f2s(gyro.gyro.y, 3), f2s(gyro.gyro.z, 3), millis());
Serial.println(buff);
logfile.println(buff);

//A delay of 30-50ms is recommended when sending adjacent Xbee packets.
delay(50);

static unsigned long millisLast = millis();

```

```

//Flush data to the log file every 120ms.
if (millis() - millisLast > 120)
{
    millisLast += 120;
    logfile.flush();
}

static unsigned long millisOne = millis();
static unsigned long millisV = millis();

//Flash the LED every 3 seconds for 50 ms.
if (millis() - millisOne < 50)
{
    digitalWrite(FET, 1);
}
else if (millis() - millisOne > 50 && millis() - millisOne < 3000)
{
    digitalWrite(FET, 0);
    millisOne += 3000;
}

//read the voltage every 1 minute and handle it if it's lower than 5.2.
if (millis() - millisV > 60000)
{
    int sensorValue = analogRead(A8);
    float V = (sensorValue / 4.092) / 10;

    sprintf(buff, "(T, %s, V, %s)", f2s(temp.temperature, 3), f2s(V, 3));
    Serial.println(buff);
    logfile.println(buff);
    logfile.flush();

    millisV += 60000;

    if (V < 5.20)
        SleepNow();
}

//Post GPS data every 300k ms, i.e. 5m
static unsigned long millisGPS = millis();

```

```

while (millis() - millisGPS > 300000)
{
    while (n_sentences != 1 && gpsSerial.available())
        displayNMEA();

    logfile.flush();
    n_sentences = 0;
    millisGPS += 300000;
}
}

//This code handles displaying the NMEA sentence.
void displayNMEA()
{
    static int i = 0;
    while (gpsSerial.available())
    {
        char ch = gpsSerial.read();
        if (ch != '\n' && i < sentenceSize)
        {
            sentence[i] = ch;
            i++;
        }
        else
        {
            sentence[i] = '\0';
            i = 0;
            //displayGPS();
            char field[20];
            getField(field, 0);
            //Serial.println(field);

            if (strcmp(field, "$GPGGA") == 0)
            {
                Serial.print('[');
                Serial.print(sentence);
                Serial.println(']');
                // delay(300);
                logfile.println(sentence);
                n_sentences++;
            }
        }
    }
}

```

```

        }
    }

void getField(char* buffer, int index)
{
    int sentencePos = 0;
    int fieldPos = 0;
    int commaCount = 0;
    while (sentencePos < sentenceSize)
    {
        if (sentence[sentencePos] == ',')
        {
            commaCount++;
            sentencePos++;
        }
        if (commaCount == index)
        {
            buffer[fieldPos] = sentence[sentencePos];
            fieldPos++;
        }
        sentencePos++;
    }
    buffer[fieldPos] = '\0';
}

//A quick function to convert a float to string for sprintf calls.
char *f2s(float f, int p)
{
    char * pBuff;
    const int iSize = 10;
    static char sBuff[iSize][20];
    static int iCount = 0;
    pBuff = sBuff[iCount];
    if (iCount >= iSize - 1) {
        iCount = 0;
    }
    else {
        iCount++;
    }
    return dtosstrf(f, 0, p, pBuff);
}

```

```

}

void SleepNow()           // Here we put the arduino to sleep.
{
  Serial.println("{Low voltage detected. Putting Arduino in sleep mode and ending data acquisition..}");
  logfile.println("Low voltage detected. Putting Arduino in sleep mode and ending data acquisition..");
  logfile.flush();
  delay(500);

  set_sleep_mode(SLEEP_MODE_PWR_DOWN);
  sleep_enable();
  sleep_mode();
}

```

ii. Receiver

```

#define SOP '<'
#define EOP '>'
bool started = false;
bool ended = false;

char inData[100];
byte index;

void setup() {
  Serial.begin(115200);
}

void loop()
{
  // Read all serial data available.
  while (Serial.available() > 0)
  {
    char inChar = Serial.read();
    if (inChar == SOP || inChar == '[' || inChar == '{' || inChar == '(')
    {
      index = 0;
      inData[index] = '\0';
      started = true;
      ended = false;
    }
  }
}

```

```

else if (inChar == EOP || inChar == ']' || inChar == ')' || inChar == ')')
{
    ended = true;
    break;
}
else
{
    if (index < 99)
    {
        inData[index] = inChar;
        index++;
        inData[index] = '\0';
    }
}
//We are most likely here because an entire packet has been received.
if (started && ended)
{
    if (inData[0] == 'L') {
        Serial.println("Low voltage detected. Arduino put in sleep mode and data acquisition ended.");
    }
    else
    {
        float values[7];
        float temp[4];
        float gps[13];
        byte index = 0;

        // Get each token.
        char *token = strtok(inData, ",");
        while (token)
        {

            if (inData[0] == '$')
                gps[index] = atof(token); //NMEA sentences begin with a $, so this one goes to the GPS index.
            else if (inData[0] == 'T')
                temp[index] = atof(token); // T denotes the temperature/voltage packet.
            else
                values[index] = atof(token); //Otherwise it's the accelerometer packet.
        }
    }
}

```

```

    index++;
    token = strtok(NULL, ","); // Parse the string.
}

if (inData[0] == '$')
{
    Serial.print(gps[1], 3);
    Serial.print(",");
    Serial.print(gps[3], 3);
    Serial.print(",");
    Serial.print(gps[5], 3);
    Serial.print(",");
    Serial.println(gps[9], 3);
}
else if (inData[0] == 'T')
{
    Serial.print("T, ");
    Serial.print(temp[1], 3);
    Serial.print(", V, ");
    Serial.println(temp[3], 3);
}
else
{
    Serial.print(values[0], 3);
    Serial.print(",");
    Serial.print(values[1], 3);
    Serial.print(",");
    Serial.print(values[2], 3);
    Serial.print(",");
    Serial.print(values[3], 3);
    Serial.print(",");
    Serial.print(values[4], 3);
    Serial.print(",");
    Serial.print(values[5], 3);
    Serial.print(",");
    Serial.println(values[6], 3);
}
}

//Reset for the next packet
started = false;

```

```

    ended = false;
    index = 0;
    inData[index] = '\0';
}
}

```

3. Structure of Data

The sampling rate of the LSM9DS0 accelerometer can be configured in the LSM9DS0 Arduino library header file and was set to 100 Hz. This data is sent through the analog-to-digital converter built-into the Arduino, which has a theoretical maximum sampling rate of 9615 Hz. The I2C bus, over which the accelerometer data is acquired, bus has a default sampling rate of 100 kHz. The highest data rate to read one block of acceleration and gyroscope data, 24 bytes (4 for each direction) is given by,

$$F = \frac{100 \text{ kHz}}{\frac{9 \text{ bits}}{\text{clock cycle}} * 24 \text{ bytes}} = 463 \text{ Hz}$$

which is well beyond our sample rate of acceleration and gyroscopic data of 100 Hz. If the sample rate is increased beyond this frequency value, e.g. 1 kHz, then the I2C clock can be set to 400 kHz. The data is sent over the XBee in the following format:

< Ax, Ay, Az, Gx, Gy, Gz, T >

Figure K-8 – Accelerometer Data Packet Format

Where Ax, Ay, Az, Gx, Gy and Gz are the acceleration components in m/s² and orientation components in degrees per second respectively and T is the time elapsed since the Arduino has been running in milliseconds. Shown below is a sample packet on the receiving end.

-0.906, 0.218, -9.934, 0.455, -0.446, -7.385, 24329

Figure K-9 – Sample Accelerometer Data Packet

Every one minute, the temperature of the container and voltage of the battery monitored by the Arduino are printed respectively. Since there are no rapid changes in these readings, there is no need to print them alongside the other packages. The packet is printed in the following format:

(T, 30.250, V, 5.74)

Figure K-10 – Sample Temperature & Voltage Data Packet

Every 30 minutes, data from the GPS module is read and the NMEA sentence is printed on the serial monitor.

To ensure the on-board SD card logs data, the first step was to ensure the SD card is detected. If it is, the following message is displayed.

SD card detected. Writing contents to Data.txt..

Figure K-11 – Arduino Message for SD Card Detected

While the following is displayed if it is missing:

No SD card detected. Data will be transmitted, but not stored.

Figure K-12 – Arduino Message for SD Card Not Detected

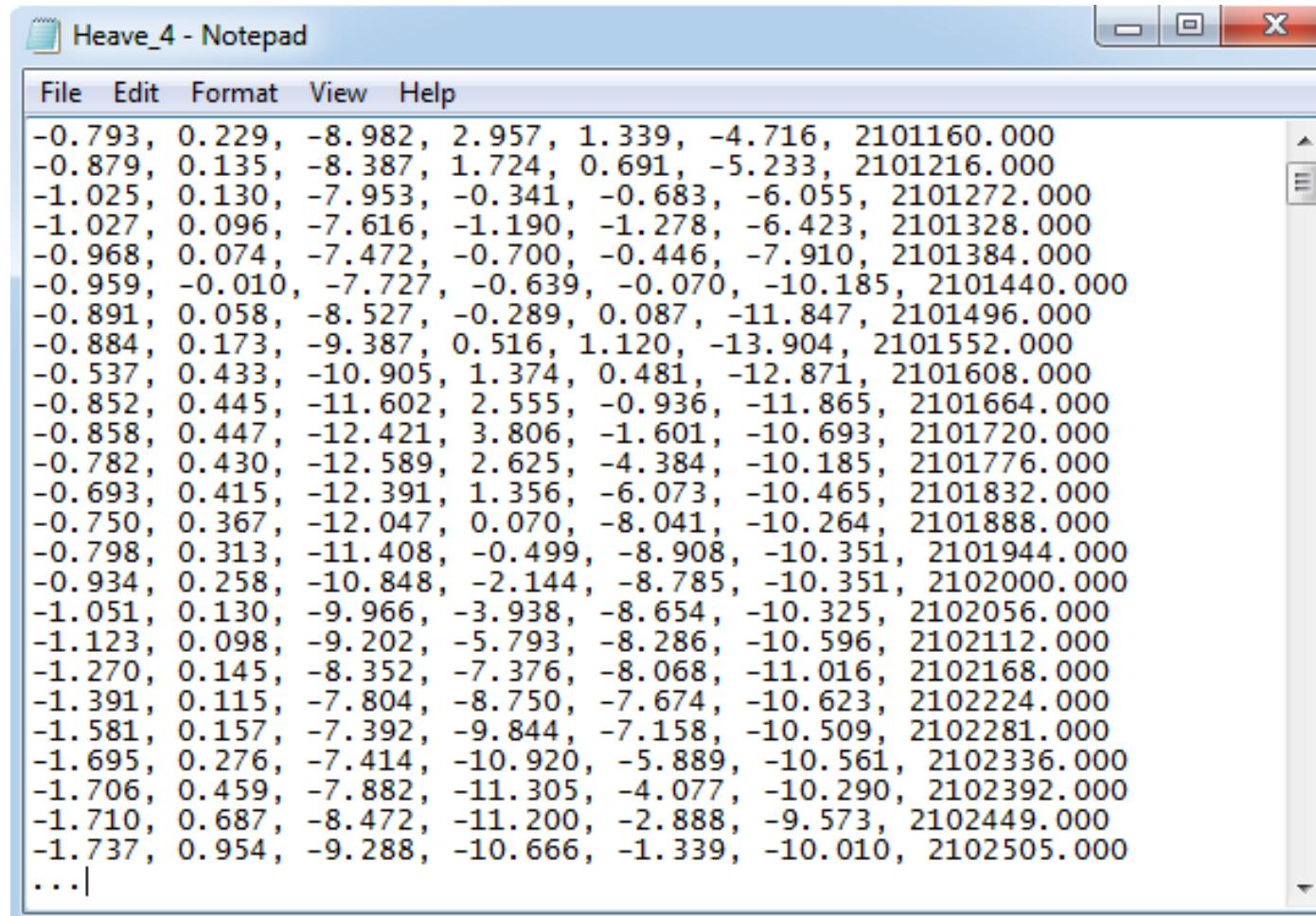
The figure below shows packets of data on the serial monitor at a baud rate of 115,200.

A screenshot of the Arduino Serial Monitor window. The window has a light blue header bar with the title "Serial Monitor". Below the header is a large text area containing a list of 20 data packets. Each packet consists of seven floating-point numbers followed by a 32-bit integer. The data is separated by commas. At the bottom of the window are several control buttons: "Autoscroll" (checked), "No line ending", "115200 baud", and "Clear output".

-0.896, 0.270, -9.934, 0.726, -0.376, -7.000, 54532
-0.900, 0.281, -9.931, 0.219, -0.718, -7.122, 54598
-0.904, 0.257, -9.944, 0.858, -0.271, -7.044, 54664
-0.885, 0.266, -9.964, 0.665, -0.175, -7.543, 54730
-0.906, 0.260, -9.978, 0.376, -0.507, -7.420, 54801
-0.878, 0.238, -9.922, 0.586, -0.262, -6.983, 54867
-0.890, 0.273, -9.932, 0.333, -0.726, -7.166, 54933
-0.900, 0.269, -9.900, 0.507, -0.123, -7.140, 55000
-0.908, 0.273, -9.925, 0.586, -0.175, -7.350, 55065
-0.907, 0.268, -9.949, 0.892, -0.201, -7.105, 55132
-0.890, 0.254, -9.950, 0.569, -0.368, -7.000, 55196
-0.889, 0.261, -9.957, 0.333, -0.683, -7.035, 55264
-0.894, 0.261, -9.904, 0.298, -0.420, -7.280, 55330
-0.896, 0.262, -9.941, 0.543, -1.076, -7.315, 55396
-0.921, 0.257, -9.940, 0.123, -0.438, -7.044, 55462
-0.921, 0.263, -9.927, 0.149, -0.420, -7.271, 55528
-0.929, 0.255, -9.924, 0.604, -0.438, -7.508, 55595

Figure K-13 – Received Packet Data on the Arduino Uno Serial Port

4. Sample Log File



The screenshot shows a Windows Notepad window titled "Heave_4 - Notepad". The menu bar includes File, Edit, Format, View, and Help. The main window displays a series of numerical values separated by commas, representing a log file. The data consists of approximately 40 rows of coordinates and a final ellipsis. The values range from -1.793 to 0.954.

Row	Column 1	Column 2	Column 3	Column 4	Column 5	Column 6	Column 7
1	-0.793	0.229	-8.982	2.957	1.339	-4.716	2101160.000
2	-0.879	0.135	-8.387	1.724	0.691	-5.233	2101216.000
3	-1.025	0.130	-7.953	-0.341	-0.683	-6.055	2101272.000
4	-1.027	0.096	-7.616	-1.190	-1.278	-6.423	2101328.000
5	-0.968	0.074	-7.472	-0.700	-0.446	-7.910	2101384.000
6	-0.959	-0.010	-7.727	-0.639	-0.070	-10.185	2101440.000
7	-0.891	0.058	-8.527	-0.289	0.087	-11.847	2101496.000
8	-0.884	0.173	-9.387	0.516	1.120	-13.904	2101552.000
9	-0.537	0.433	-10.905	1.374	0.481	-12.871	2101608.000
10	-0.852	0.445	-11.602	2.555	-0.936	-11.865	2101664.000
11	-0.858	0.447	-12.421	3.806	-1.601	-10.693	2101720.000
12	-0.782	0.430	-12.589	2.625	-4.384	-10.185	2101776.000
13	-0.693	0.415	-12.391	1.356	-6.073	-10.465	2101832.000
14	-0.750	0.367	-12.047	0.070	-8.041	-10.264	2101888.000
15	-0.798	0.313	-11.408	-0.499	-8.908	-10.351	2101944.000
16	-0.934	0.258	-10.848	-2.144	-8.785	-10.351	2102000.000
17	-1.051	0.130	-9.966	-3.938	-8.654	-10.325	2102056.000
18	-1.123	0.098	-9.202	-5.793	-8.286	-10.596	2102112.000
19	-1.270	0.145	-8.352	-7.376	-8.068	-11.016	2102168.000
20	-1.391	0.115	-7.804	-8.750	-7.674	-10.623	2102224.000
21	-1.581	0.157	-7.392	-9.844	-7.158	-10.509	2102281.000
22	-1.695	0.276	-7.414	-10.920	-5.889	-10.561	2102336.000
23	-1.706	0.459	-7.882	-11.305	-4.077	-10.290	2102392.000
24	-1.710	0.687	-8.472	-11.200	-2.888	-9.573	2102449.000
25	-1.737	0.954	-9.288	-10.666	-1.339	-10.010	2102505.000
26	...						

Figure K-14 – Sample Log File on SD Card

Appendix L

Acceleration Graphs

Heave 1

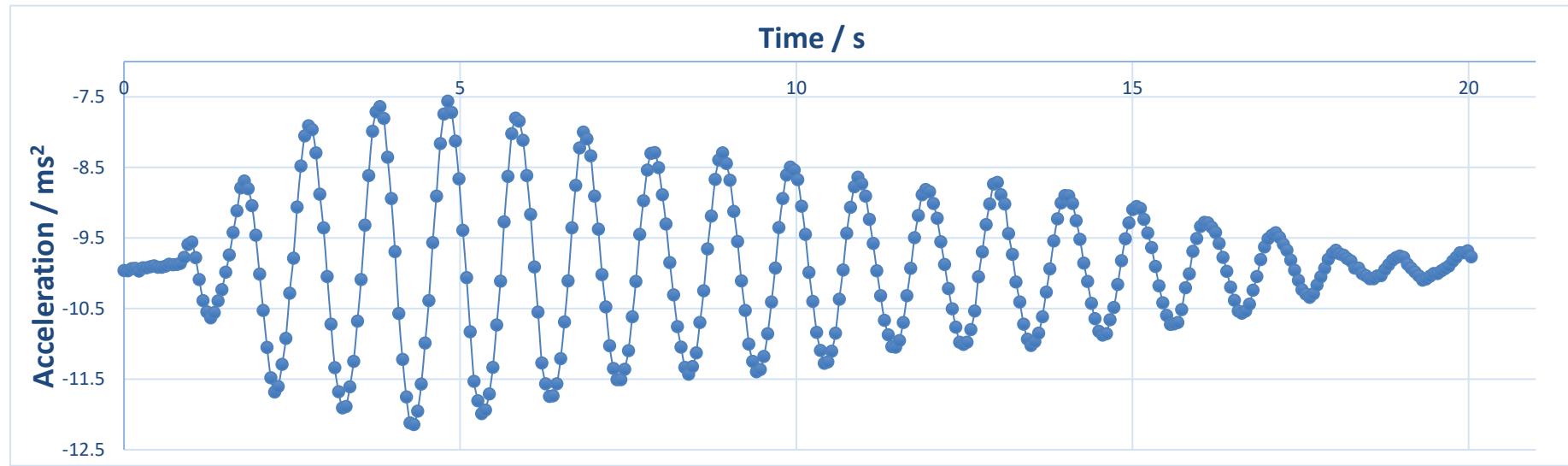


Figure L-1 – Heave 1 Acceleration Graph

Recorded Time Interval (s)	20.0
Number of Points	359
Average Time Interval (ms)	56
Max. Acceleration (m/s²)	-7.8
Min. Acceleration (m/s²)	-12.1
Mean Acceleration (m/s²)	-9.92
Average Time Period (s)	0.83
Average Frequency (Hz)	1.21

Table L.1 – Heave 1 Acceleration Characteristics

Heave 2

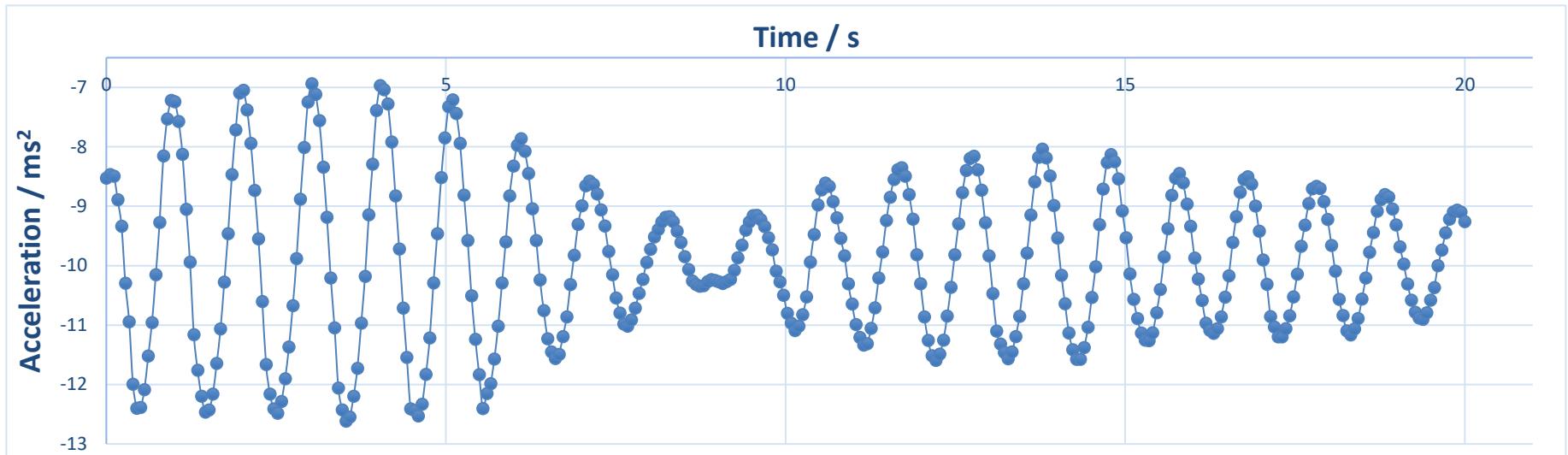


Figure L-2 – Heave 2 Acceleration Graph

Recorded Time Interval (s)	20.0
Number of Points	358
Average Time Interval (ms)	56
Max. Acceleration (m/s²)	-6.9
Min. Acceleration (m/s²)	-12.6
Mean Acceleration (m/s²)	-9.93
Average Time Period (s)	0.99
Average Frequency (Hz)	1.00

Table L.2 – Heave 2 Acceleration Characteristics

Heave 3

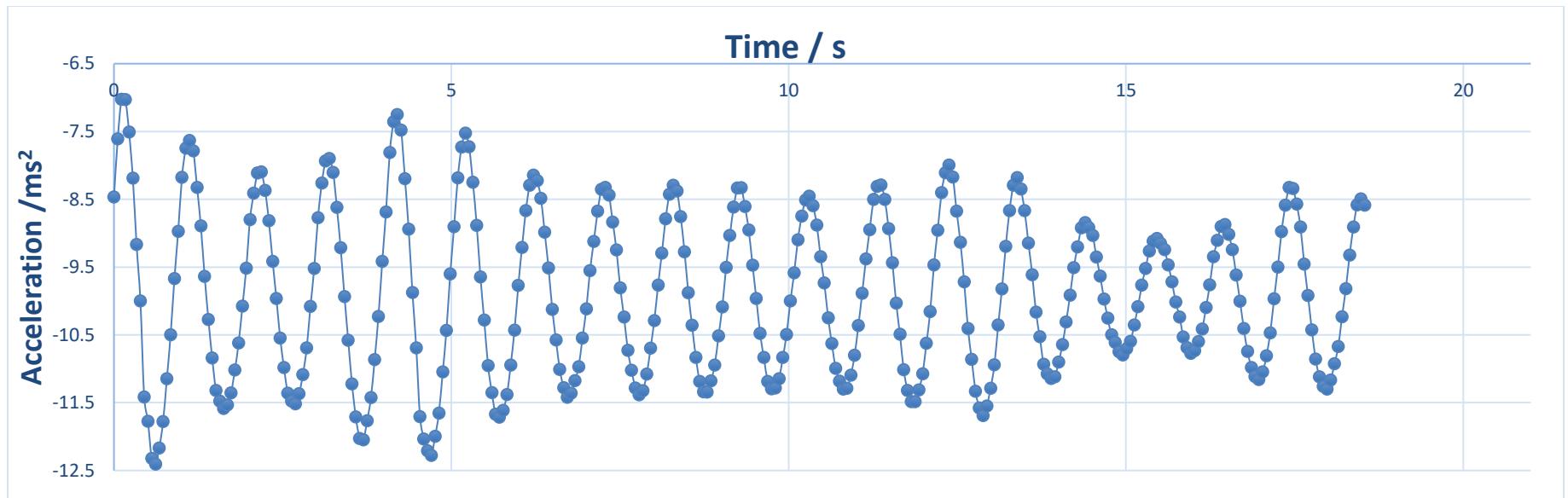


Figure L-3 – Heave 3 Acceleration Graph

Recorded Time Interval (s)	18.6
Number of Points	332
Average Time Interval (ms)	56
Max. Acceleration (m/s²)	-7.0
Min. Acceleration (m/s²)	-12.4
Mean Acceleration (m/s²)	-9.90
Average Time Period (s)	1.02
Average Frequency (Hz)	0.97

Table L.3 – Heave 3 Acceleration Characteristics

Heave 4

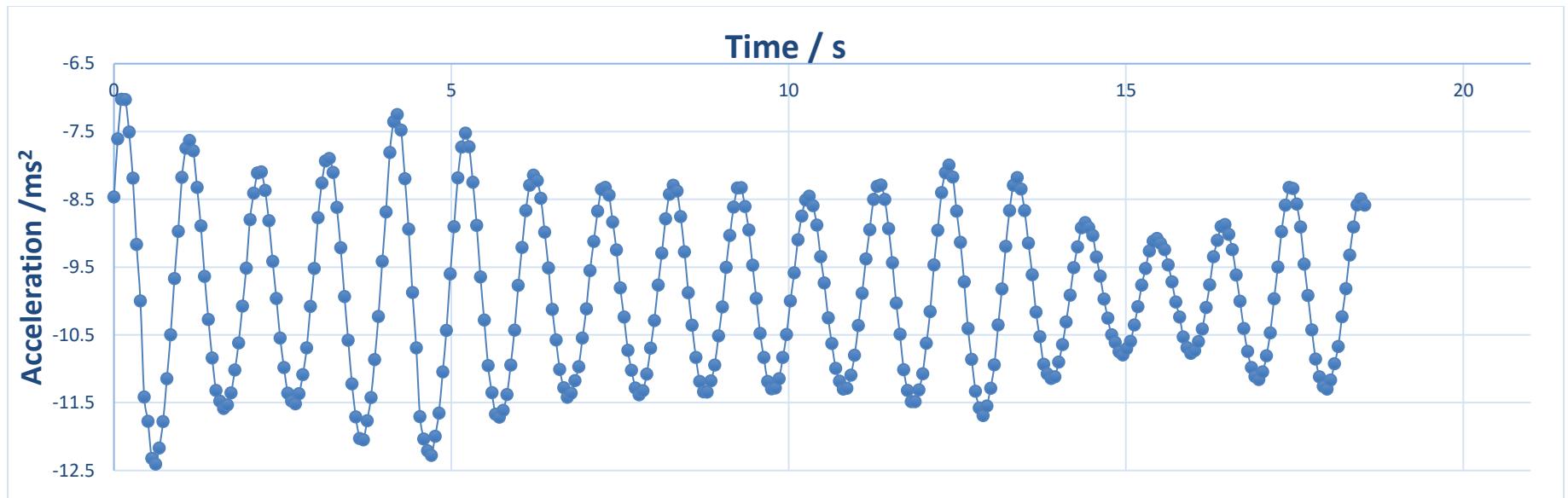


Figure L-4 – Heave 4 Acceleration Graph

Recorded Time Interval (s)	20.0
Number of Points	376
Average Time Interval (ms)	56
Max. Acceleration (m/s²)	-7.0
Min. Acceleration (m/s²)	-12.3
Mean Acceleration (m/s²)	-9.90
Average Time Period (s)	0.99
Average Frequency (Hz)	1.00

Table L.4 – Heave 4 Acceleration Characteristics

Appendix M

References

- [1] <https://www.whoi.edu>, ‘The 11th MTS Buoy Workshop’, 2016. [Online]. Available: <https://www.whoi.edu/buoyworkshop/2016/call-for-speakers.html>. [Accessed: 15- Sep- 2016]
- [2] Raspberry Pi Data Sheet, <https://www.adafruit.com/datasheets/pi-specs.pdf>
- [3] Arduino Mega 2560 Technical Specifications, <http://www.arduino.cc/en/Main/ArduinoBoardMega2560>
- [4] Intel Edison Data Sheet, https://www.intel.com/content/dam/support/us/en/documents/edison/sb/edison-module_HG_331189.pdf
- [5] Parallax Propeller Data Sheet, https://cdn.sparkfun.com/datasheets/Dev/Propeller/Propeller-P8X32A-Datasheet-v1.4.0_1.pdf
- [6] Bluetooth Range, <https://www.sans.edu/cyber-research/security-laboratory/article/bluetooth>
- [7] XBee Module Datasheet, <https://www.sparkfun.com/datasheets/Wireless/Zigbee/XBee-Datasheet.pdf>
- [8] WiFi Module Datasheet, https://cdn-shop.adafruit.com/datasheets/ESP8266_Specifications_English.pdf
- [9] Distance Testing, <http://hackaday.com/2014/09/26/esp8266-distance-testing/>
- [10] nRF2401 Datasheet, https://www.sparkfun.com/datasheets/RF/nRF2401rev1_1.pdf
- [11] Ashby, Michael.F., ‘Materials Selection in Mechanical Design’, 4th Ed., Butterworth-Heinemann, Oxford, United Kingdom, Chapter 4. Figures of Yield Strength and Fracture Toughness, 2010.

- [12] Hearn, E. J., '*An Introduction to the Mechanics of Elastic and Plastic Deformation of Solids and Structural Materials*', 3rd Ed., Butterworth-Heinemann, Oxford, United Kingdom, Chapter 7. Circular Plates and Diaphragms, 1997.
- [13] Fontana, M. G., '*Corrosion Engineering*', 3rd Ed., McGraw Hill Education, New York, United States, Chapter 6. Corrosion Prevention, 2005.
- [14] Parker., '*Parker O-Ring Handbook*', Parker Seal Group, O-Ring Division, 2007.