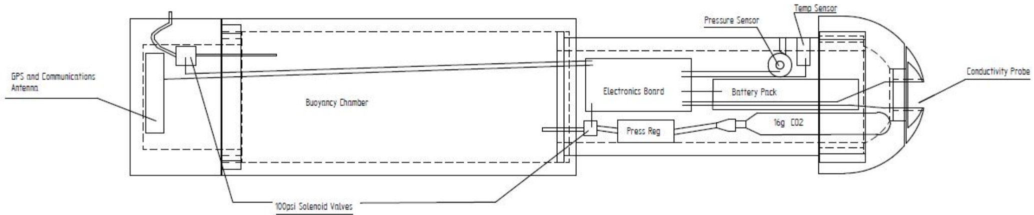
NAVSEA/FIT - ECS Unit



by
Richard Francis Paradis

Bachelor of Science
Ocean Engineering
Florida Institute of Technology
2011

A thesis
submitted to the College of Engineering at Florida Institute of Technology
in partial fulfillment of the requirements
for the degree of

Master of Science
in
Department of Marine and Environmental Science - Ocean Engineering

Melbourne, Florida
May, 2013

We the undersigned committee
hereby approve the attached thesis

NAVSEA/FIT - ECS Unit

by
Richard Francis Paradis

Stephen Wood, Ph.D., P.E.
Associate Professor
Ocean Engineering Program Chair
Marine and Environmental Systems
Committee Chair

Hector Gutierrez, Ph.D., P.E.
Associate Professor
Mechanical and Aerospace
Engineering

George A. Maul, Ph.D.
Professor and Department Chair
Marine and Environmental Systems

Abstract

NAVSEA/FIT - ECS Unit

by

Richard Francis Paradis

Major Advisor: Stephen Wood, Ph.D., P.E.

The Conductivity, Temperature, and Depth (CTD) sensor is one of the most used instruments in the Oceanographic field. These devices are the number one, most important sensor for any research of the ocean depths. CTD measurements are found in every marine related institute and navy throughout the world because they are used to produce the salinity profile for the area of the ocean under investigation. In order to deploy large numbers of CTD sensors, an inexpensive version is needed, especially if used as a one-time (throw-away) system. Oceanographic institutes and navies require instruments such as these that are reliable, accurate, precise and inexpensive.

The United States Navy (Naval Surface Warfare Center Panama City Division (NAVSEA-PCD)) is highly interested in the development of a low cost disposable CTD that can be easily deployed rapidly to obtain measurements throughout the earths oceans. NAVSEA-PCD approached Florida Institute of Technology with five (5) specific specifications for the device (ECS Unit Sensor): 1) Its weight had to be under five pounds maximum, but ideally as close to one to two pounds as possible; 2) Its dimensions must be no bigger than ten inches in height and three inches in diameter; 3) It must be reasonably priced to be expendable; 4) It must have wireless communication; and 5) It only has to record the top ten meters of the water column.

This thesis describes the design, development, construction and testing of the ECS Unit Sensor, which when deployed, takes multiple snapshots of the water profile and sends its data out via a satellite communications chip. The

ECS Unit was tested against other CTDs for accuracy. When compared to the other expendable CTDs the Expendable Conductivity Sensor Unit (ECS Unit), along with being much more economical, is able to provide more snapshots than any other expendable CTD instrument currently available on the market. The CTD instrument (ECS Unit Sensor) developed in this thesis costs around $982.00 per unit to make.

Table of Contents

List of Figures

List of Tables

Acronyms

CO_2	Carbon Di-Oxide
AUV	Autonomous Underwater Vehicle
AXCTD	Aerial eXpendable CTD
CTD	Conductivity, Temperature, and Depth
ECS Unit	Expendable Conductivity Sensor Unit
FIT	Florida Institute of Technology
FM Signal	Frequency Modulated Signal
GPS	Global Positioning System
HROV	Hybrid Remotely Operated Vehicle
NAVSEA-PCD	Naval Surface Warfare Center Panama City Division
NMEA	National Marine Electronics Association
ppt	Parts Per Thousand

psi	pounds per square inch
PSS	Practical Salinity Scale of 1978
REMUS	Remote Environmental Monitoring UnitS
ROV	Remotely Operated Vehicle
S’	Seimens
SONAR	SOund Navigation And Ranging
TACSAT	Tactical Satellite
WIFI	Wireless Fidelity
XCTD	eXpendable CTD

List of Equations

Acknowledgment

I could not have completed this project without the help from Mathew Jordan. Matt was there every step of the way and brought a great deal of embedded programming expertise to the project along with a vast knowledge of circuit designing. Thank you Matt for all your help.

Also a special thanks to:

Dr. Stephen Wood
Dr. George Maul
Dr. Hector Gutierrez
Dr. Charles Bostater
Bill Battin
Jim Tryzbiak
Bill Bailey
Ed Paradis, Sr.
Alexandra Gottschall
Joshua Lappen
Sean Pagliari
Jess Haig
Tony Bond
Robert Gibson

Preface

This whole project started back in May 2011 when I had just completed my first semester in the Ocean Engineering masters program at Florida Institute of Technology (FIT). I had applied to Naval Surface Warfare Center Panama City Division (NAVSEA-PCD) for an internship along with two other students from FIT - Robert Pagliari and Jessica Haig. When we started our summer internship with NAVSEA-PCD, the three of us were given the task of designing an expendable Conductivity, Temperature, and Depth (CTD). We bounced around a few designs and then finally settled on a telescoping unit that could make several dives before it became no longer functional. Some of the folks in at NAVSEA-PCD were unsure if the design would even be buoyant enough to hold up its own weight; so we constructed a crude, proof of concept, to see if it would work. Our crude design consisted of an empty tin can that had the same volume as the designed buoyancy chamber for the disposable CTD units design and weights were taped to the bottom of it to simulate 150% of what the designed weight was to be. It floated. Elated at our findings, we showed our results to those people who did not believe it would actually work. We, quickly moved onto the next phase of the project; making a proof of concept for the conductivity sensor. For this we had designed a four probe sensor (which I later scrapped due to its high complexity and low preciseness) and tested it out. It worked very well part of the time, yet we could not get multiple readings to stay within a 10% error range. With the thought of being able to improving upon it later, we moved onto designing the proof of concept for the buoyancy system that would control its movement in the water column. We had 3D printed the housing of the actual expendable CTD unit that we had designed in Solid Edge to get an idea of how each part would fit inside it. So we used that to test the buoyancy system as well. It was found out that it was not waterproofed enough and had developed a small leak, which caused that test to fail, but it was at this point that our internship at NAVSEA-PCD was over and time to return back to FIT to begin a new semester of classes. However I was disappointed at not being able to complete the task given me. so after speaking with Tony Bond at NAVSEA-PCD as well as with Dr. Wood at FIT, it was agreed I would continue working on it for my thesis.

Chapter 1

Introduction

The CTD sensor is one of the most used instruments in the Oceanographic field, and are the most important sensor for any research or study being done on the ocean. CTD measurements are found in every marine related institute and navy throughout the world because they are used to produce the salinity profile for the area of the ocean under investigation. The salinity profile is important because it gives insight into the varying water densities in the ocean. This is a key part in understanding deep ocean currents and the ocean's acoustical properties. While research of the deep sea currents, which are driven by the difference in density throughout a column of water, are important to oceanographers; the Navy is more interested in the acoustical properties of the ocean. These play an integral role for SOund Navigation And Ranging (SONAR) and are used to help determine the vibration signatures for underwater vehicles.

The cost of the standard CTD instruments, whether expendable or not, is very expensive. In order to deploy large numbers of CTD sensors, an inexpensive version is needed, especially if used as a one-time (throw-away) system. This thesis covers the design and testing of an alternative expendable CTD (Expendable Conductivity Sensor Unit (ECS Unit)) to what is currently available on the market (which will be discussed in section 2.1.1). The ECS Unit was designed around NAVSEA-PCD's five (5) specific specifications for this device:

1) Its weight had to be under five pounds maximum, but ideally as close to one to two pounds as possible; 2) Its dimensions must be no bigger than ten inches in height and three inches in diameter; 3) It must be reasonably priced to be expendable; 4) It must have wireless communication; and 5) It only has to record the top ten meters of the water column.

This thesis describes the design, development, construction and testing of the ECS Unit Sensor, which when deployed, takes multiple snapshots of the water profile and sends its data out via a satellite communications chip. Currently the expendable CTD devices on the market today are only capable of a single snapshot of the water profile, per unit deployed. When the ECS Unit was compared to these CTDs, it was found to me more economical instrument. This was due to the fact that while they can only take a single snapshot per unit, the ECS Unit has the ability to take multiple snapshots. The CTD instrument (ECS Unit Sensor) developed in this thesis costs about $1,500 per unit to make.

1.1 Introduction to CTD's

Scientists and Oceanographers use CTD's to get the salinity profile of a water column in the ocean. These sensors give scientists a precise and comprehensive charting of the distribution and variation of water temperature, salinity, and density which helps to understand how the oceans affect life [15]. Theses properties help oceanographers with their studies of the ocean, but more importantly for this thesis, these properties also help the oceanographers and Navy scientists understand how the density variations will affect the acoustical properties of the water.

Generally CTDs are very large, very expensive, and are tethered to a support vehicle, usually a boat lowering it down into the water column. However, CTDs can be smaller, like the units used in an Autonomous Underwater Vehicle (AUV) or glider, but unlike their size, their cost increases and these smaller versions can get pretty expensive. So because of their price,CTDs are not expendable. There are two exceptions though, the XCTD and AXCTD. While these were designed to be throw away, the cost of these CTDs prevent them from being used with most oceanographic expeditions, whose budgets are tight and can not spare the cost of a single use CTD.

1.2 CTDs - How they Work

The main sensors that make up a standard CTD package include a temperature sensor, conductivity meter, and pressure sensor. The temperature sensor of course gives us temperature measurements at the instruments location in the water column, and those readings help determine the salinity reading. The conductivity meter measures conductance, which is the amount of electrical current that can pass through the water; and with a few calculations and some input from the temperature probe, salinity can be determined. The pressure sensor of course measures pressure, which is a function of depth. So ultimately as the instrument is lowered into the water column it takes readings using the different sensors and through on-board calculations and data processing it will give us temperature and salinity readings with respect to depth. These measurements give us insight into the properties of sea water [15].

1.3 What is Salinity

Salinity provides the measurement of the amount of salt by mass, to a unit mass of seawater [19]. Understanding why the sea is salty begins with the knowledge of the water cycle and its effects on the ocean's physical states. Water in liquid form dissolves rocks and sediments and absorbs the emissions from volcanoes and hydrothermal vents, creating a complex solution of minerals and salts in the oceans around the globe. Yet, when water is in the form of vapor and ice, it is incompatible with salts and is practically salt free. Since most of the global evaporation and precipitation occurs over the ocean, the ocean surface salinity is very important to understand how fresh water input and output affects the ocean dynamics and physical properties[22]. One of the most important physical properties for the Navy is the density of the ocean water. This directly affects the acoustical properties of the ocean with which the speed of sound is calculated along with the sound wave reflection/refraction/diffraction properties. The most important of these properties though is the speed of sound "c" calculated using equation 1.1 provided by Hermin Medwin using the temperature "T", depth "z", and salinity of the water "S" [20].

$$c = 1449.2 + 4.6T - 0.055T^2 + 0.00029T^3 + (1.34 - 0.010T)(S - 35) + 0.16z \quad (1.1)$$

Salinity plays a key role in the health of various ecosystems around the ocean, like coral reefs, slight changes in temperature and salinity can wreak havoc on coral reefs [17]. Oceanographers are also interested in the salinity of the ocean, because along with water temperature, it determines the density of the ocean water. The salinity profiles give a snapshot of the density variations in the ocean at that location, and those profiles are important to the oceanographers because as the ocean water cools or the salinity increases, the ocean water becomes either more or less dense. The change in water density drives the ocean circulation at greater depths then wind driven circulation near the surface [18]. These profiles are extremely important aspect in understanding the ocean so oceanographers in the science and naval community all use these measurements in great quantities. By keeping track of the oceans surface salinity/density, scientists can monitor the fluctuations in the water cycle and get a better understanding of how the ocean interacts with various environments on a global scale [22].

To measure salinity, there are a few calculations that must be done using the following equations provided by Bibby Scientific [16] . First of all, a probe with a set cell constant $\boldsymbol{K}$ will have to be known, K can be calculated by dividing the Distance between the probes $\boldsymbol{D}$ by the Surface Area of the Probes $\boldsymbol{A}$:

$$\boldsymbol{K} = \boldsymbol{D}/\boldsymbol{A} \quad (1.2)$$

and by passing a known current and voltage across the probe while submerged in the salt water a conductance reading can be measured in units of Seimens (S'). To get Conductance $\boldsymbol{G}$, divide the electrical current $\boldsymbol{I}$, by the known voltage $\boldsymbol{V}$:

$$\boldsymbol{G} = \boldsymbol{I}/\boldsymbol{V} \tag{1.3}$$

With the conductance known and the probe cell constant known, conductivity $\boldsymbol{C}$ can be calculated by dividing the conductance by the probe cell constant $\boldsymbol{K}$:

$$\boldsymbol{C} = \boldsymbol{G}(\boldsymbol{K}) \tag{1.4}$$

Once conductivity is calculated it will then need to be converted into Resistivity $\boldsymbol{R}$, which is the reciprocal of conductivity $\boldsymbol{C}$.

$$\boldsymbol{R} = 1/\boldsymbol{C} \tag{1.5}$$

Now with the Resistivity in hand, along with a temperature reading, salinity $\boldsymbol{S}$ can be calculated by using the Practical Salinity Scale of 1978 (PSS) equation discussed by Edward Lewis [19] (see equation 1.5). This equation has some additional variables of its own though. It requires a temperature reading along with the unity $\boldsymbol{k}$ of the CTD instrument. The unity can be determined by taking the conductivity value of a known solution and comparing it to the conductivity reading of that solution. This ratio will then be the last variable used in the equation of the PSS [16]. With the combination of those variables and a few given constants, the PSS equation can be calculated (please see equation 1.6 on the following page).

$$\boldsymbol{S} = a_0 + a_1 R_T^{1/2} + a_2 R_T + a_3 R_T^{3/2} + a_4 R_T^2 + a_5 R_T^{5/2} +$$

$$\frac{(T-15)}{1+k(T-15}(+b_0 + b_1 R_T^{1/2} + b_2 R_T + b_3 R_T^{3/2} + b_4 R_T^2 + b_5 R_T^{5/2}) \tag{1.6}$$

where:

a_0 = 0.0080

a_1 = -0.1692

a_2 = 25.3851

a_3 = 14.0941

a_4 = -7.0261

a_5 = 2.7081

k is the sensors measure of unity

R_T is the resistivity of the water sample at temperature

T is the temperature of the water sample

b_0 = 0.0005

b_1 = -0.0056

b_2 = -0.0066

b_3 = -0.0375

b_4 = 0.0636

b_5 = -0.0144

Equation 1.6 is valid for temperatures between -2C and 35C and salinity between 2 and 42 [19]. Although this equation is good for a wide salinity range, the probe measuring the conductivity of the sea water must have the appropriate cell constant if the instrument cannot regulate the amount of current it can pass across the probes . The cell constant of saltwater should be 10, for brackish water 1, and for fresh water 0.1 cell constant [16]. Since the ECS Unit was designed for saltwater, the cell constant that was used was the $\boldsymbol{k}$=10 constant.

Chapter 2

Background

2.1 CTD Types

CTDs come in all shapes and sizes, ranging anywhere from a casting CTD (which is the most basic and popular type), to autonomous Argo Floats, or even the expendable eXpendable CTD (XCTD)s. CTD casting methods already in existence include ship assisted casting/manual casting and remote sensing. Ship assisted casts are when manual input is needed to complete a CTD cast (the lowering and raising of the CTD through the water column). For remote sensing, the operation of the CTD is completed through an AUV, Remotely Operated Vehicle (ROV), or glider without the assistance or presence of the scientist.

Most ship assisted CTDs are very large. Usually the measurements these CTD units record are either sent back via a communications cable or logged on a memory chip and requires a cable to be plugged in to the unit in order to download the measurements for analysis. The shipboard CTD consists of small probes attached to a large metal rosette wheel which can hold other instruments as well. If a communications cable is used, scientists can observe the water properties in real time. A standard CTD cast, depending on water depth,

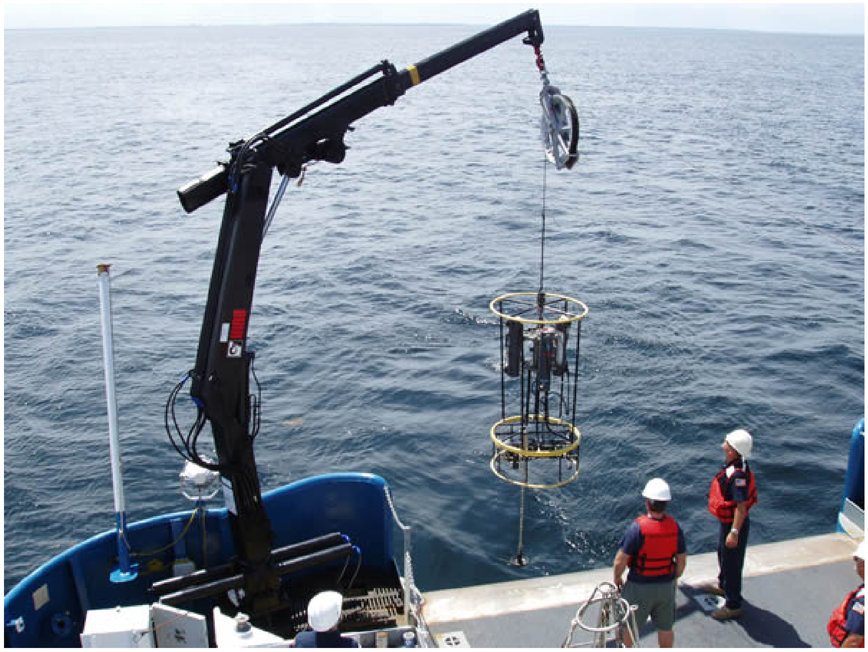


Figure 2.1: Ship Casting: [1]

usually requires a couple of hours to collect a complete set of data. Small, low-powered CTD sensors are used on expendable CTDs or autonomous platforms such as a profiling floats, AUV's, Expendable or moored profilers [15].

2.1.1 AXCTD/XCTD

The XCTD by Lockheed Martin - produced by Sippican, Inc. can be seen below in figure 2.2. The XCTD is launched out of a specially designed apparatus (hand held or hull mounted) over the side of a vessel, the sensor itself contains a reel of a micro wire which is connected to the launching apparatus, this micro wire is spooled out on the probe end and the launching apparatus end to prevent the wire from binding or kinking while it descends through the water column. The launching apparatus is connected to a specialized data acquisition system (the MK21 DAQ System)designed for the XCTD via a data cable. The data acquisition system records the data at a rate of 800Hz from the probe while the micro wire continues to unspool, and upon reaching the extent of micro-wire length the wire breaks (or is cut) and the data logging ceases, therefore the unit can only be used once.

Figure 2.2: XCTD: [4]

Remote sensing CTDs are usually either an Unmanned Underwater Vehicle (UUV) with on board CTD packages incorporated into its system, or expendable units that require a support vehicle nearby to receive the data. The autonomous vehicles can send out data packets to a central computer when it surfaces, or can record measurements onto a memory chip and be downloaded when the vehicle is retrieved. These UUV systems are very expensive and need to undergo maintenance and repairs each time before being sent back out. The type of expendable CTDs used in this method of data collecting is usually the AXCTD which is very similar to the XCTD and also made by Sippican, Inc. and can be seen in figure 2.3.

The Aerial eXpendable CTD (AXCTD) is launched from a plane or AUV and splashes down in the targeted water body below the aircraft. The unit floats on the surface and uses a wireless transmitter to send the data back to the aircraft to a computer to be processed and saved. The floating unit serves a similar role of the XCTDs launching apparatus. The actual sensor gets dropped from the floating unit and a small micro wire spool on the sensor is connected

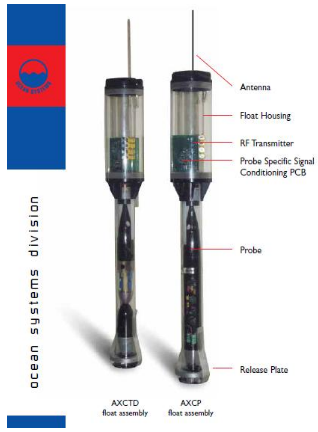


Figure 2.3: AXCTD: [4]

to the floating unit which relays the data out to the aerial support vehicle. The data is sent out as a Frequency Modulated Signal (FM Signal). When the AXCTD sensor reaches its maximum depth the micro wire breaks, so this unit can only be used once as well.

2.1.2 Argos Floats

The Argos Floats are a type of profiling float that have a complementary relationship with the Jason satellite altimeter mission. This relationship gave birth to its name of Argo, the ship in Greek mythology from the tale of Jason and the Argonauts. The large global float array with the Jason satellite give real time data to computer models to help forecast the ocean climate. There are about 3000 individual floats located in the oceans around the world that provide about 100,000 temperature/salinity (T/S) profiles (see figure 2.4) and velocity measurements per year to the computer models [25].

The Floats cycle to 2000m depth every 10 days. On board the Argos Float, there is a pump that pushes fluid into an external bladder to make it positively buoyant. It takes about 6 hours to reach the surface, all the while it takes measurements of temperature and salinity. Once it reaches the surface, satellites determine the position of the float, and the float transmits its data through the

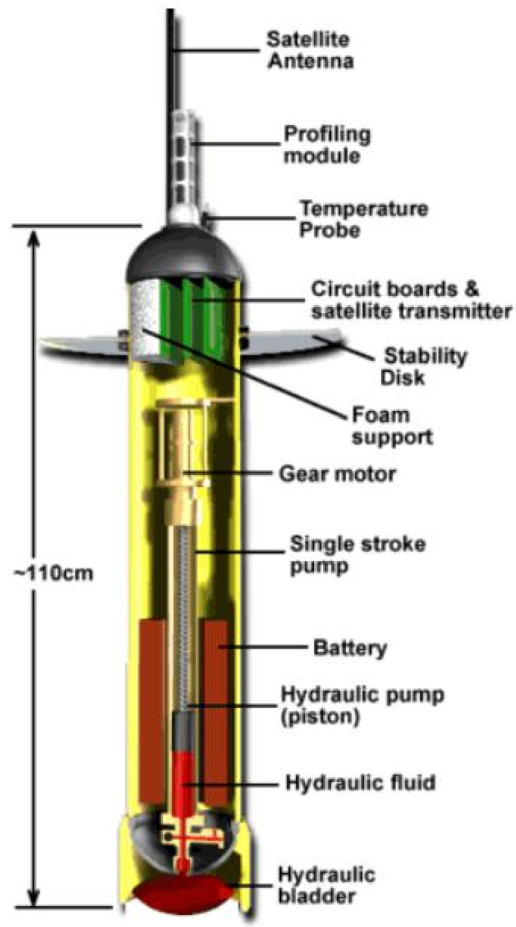


Figure 2.4: Argos Float [2]

satellites back to the computer models. After the data is sent the bladder gets deflated and the float sinks back to its predetermined depth to drift until the cycle is repeated. Each of the Argos Floats are designed to travel up in down in the water column 150 times, which should give each float between 4 to 5 years for a lifespan [25].

2.1.3 AUV or Glider CTDS

AUV's and gliders are used in every ocean around the world. AUV's actively roam around the oceans working on missions that scientists have programmed into them, and can be out on those missions for days to weeks at a time. AUV's like the Remote Environmental Monitoring UnitS (REMUS) have been used in all kinds of missions including some for the US Navy. Other AUV's can be highly specialized, and can double as a ROV, like the Nereus (see figure 2.5 below) which is an Hybrid Remotely Operated Vehicle (HROV). [12]

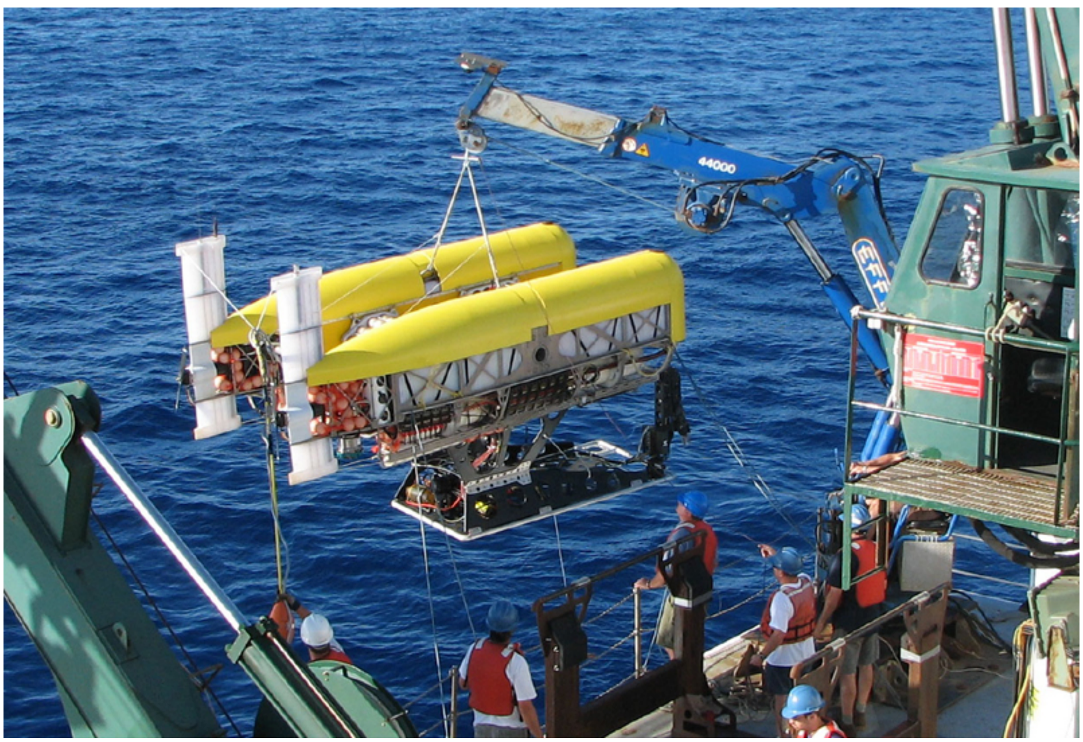


Figure 2.5: Nereus: [12]

Gliders are similar to AUV's, but generally are much smaller and use considerably less power. Rather than using thrusters to push them on their way, they dynamically adjust their buoyancy and convert the upwards or downwards motion, from that change in buoyancy, to forward motion with the pitch of their wings just like a plane would. The only difference is that the plane usually stays level while the the underwater gliders center of gravity gets changed thus changing the angle of the glider. The Slocum glider (named after the man who first sailed around the world alone, Joshua Slocum) is a prime example (seen in figure 2.6), and uses the thermal gradient in the ocean to adjust its buoyancy.

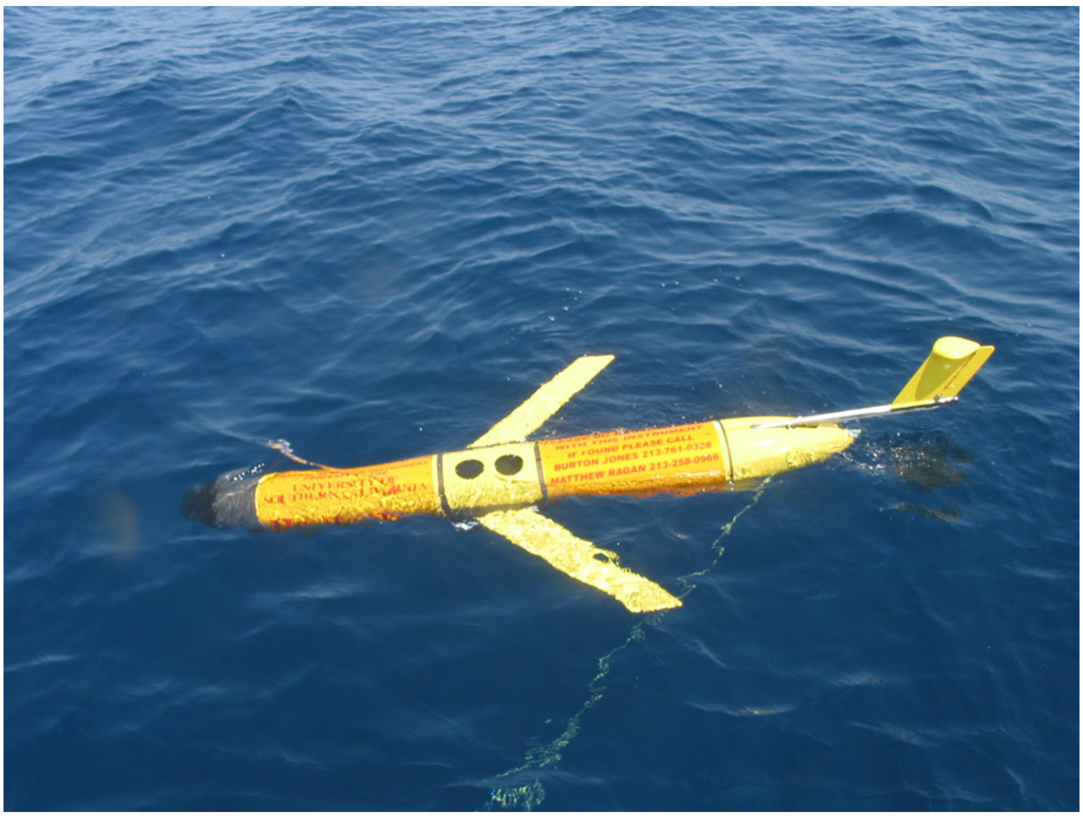


Figure 2.6: Slocum: [12]

AUV's typically have a higher payload capacity and therefore can carry higher grade CTDs then what gliders can. For example the SBE 49 Fast-CAT CTD from Sea-Bird Electronics, generally used in AUV's like the Bluefin Robotics AUV's, has a data logging rate of 16Hz. This CTD, depending on the model, has depth ratings up to 7000m; and has a resolution of 0.00005(S'/m).

It is powered from the AUV battery bank. Whereas gliders, like the Slocom Glider, have to be more mindful of their payload weight and therefore cannot carry such high end CTDs. Generally gliders tend to use CTDs like the Glider Payload CTD from SEA-Bird Electronics which has a 1Hz data logging rate and a depth rating up to 1500m; and has a resolution of 0.0003 Seimens per Meter (S'/m). Gliders do not have enough power to supply to the CTD so this one requires its own battery pack, which can power it for up to 45 days. [13]

2.1.4 Moored Profilers

A Moored Profiler is a CTD that travels up and down the water column just like the other CTDs, but rather then be free floating it is attached to a sub surface mooring and uses a traction motor to travel up and down the mooring line. The sub-surface mooring can stretches down to to the seafloor (turn page to see figure 2.7) and can be used anywhere between shallow coastal waters to the deep open ocean waters. The profiler needs a clear run of cable to transverse across, this poses a problem for all the other instrumentation being deployed on the mooring as well. This is usually taken care of by deploying a second sub-surface mooring just to hold the the other instruments.

However this can be extremely costly, and for smaller budgets, all the instrumentation needs to be attached to a single mooring. The best way around this issue is to confine the other instrumentation in groups, and in between groups, place a profiler. Running multiple profilers still allows a full transverse of the water column while providing other readings as well for selected depths. Usually these profilers have enough battery power to travel around one million meters up and down the cable. This can last up to a year before needing to be collected, cleaned, and recharged. This CTD profiler stores its data on-board for the duration of its deployment and is offloaded when the instrument is recovered. [14]

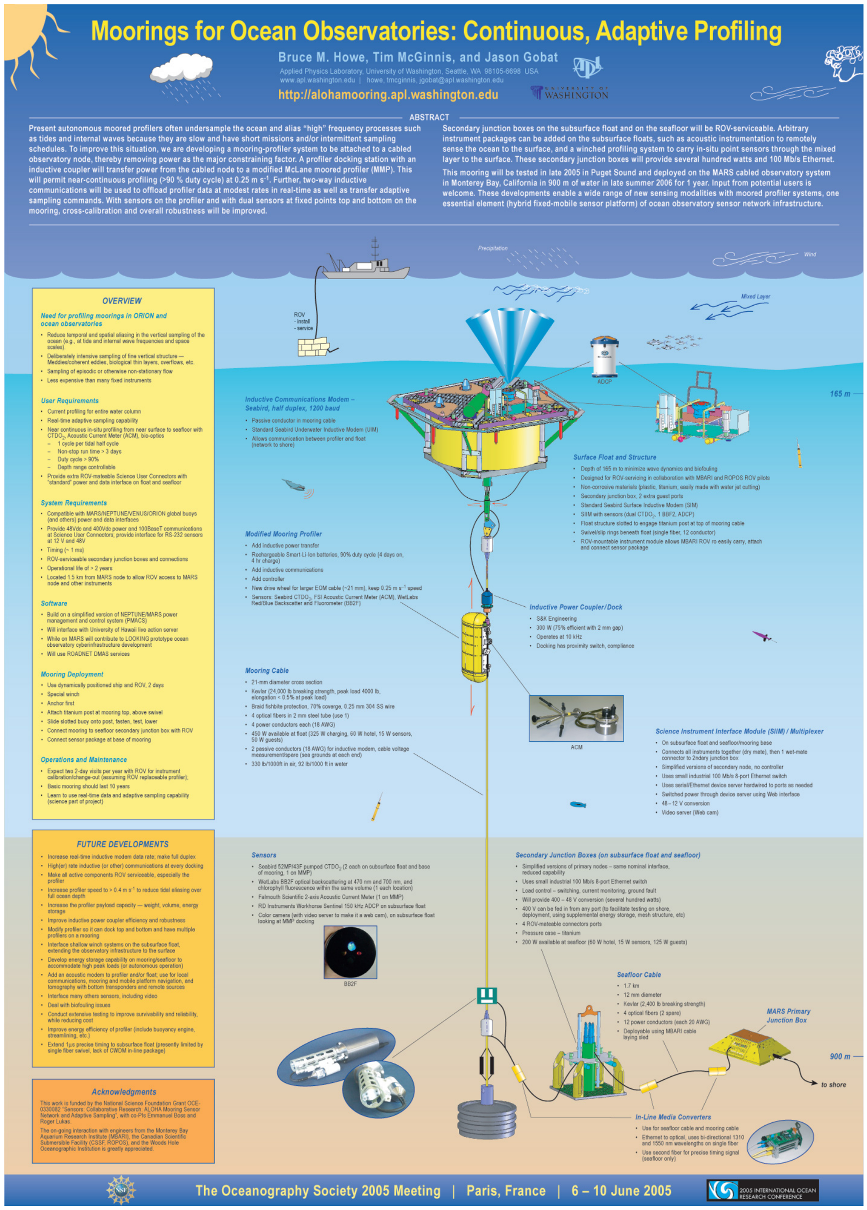


Figure 2.7: Sub Surface Moored Profiler: [6]

2.2 Buoyancy Control

Buoyancy control is the ability to maintain negative, neutral, or positive buoyancy to move through a water column; and stems all the way back to Archimedes eureka moment in the bath tub - water displacement. Buoyancy control is as simple as adjusting the water displacement/density of the platform on which the CTD unit is mounted. For AUV's their buoyancy control comes from a neutral buoyancy design and through the use of their thrusters they can determine its position in the water column. Gliders control their buoyancy by either adjusting an inflatable bladder or by manipulating materials that can adjust their density such as thermal waxes and then use the pitch of their wings to control its ascent/descent in the water column [12]. The Argos Float changes its density by pumping fluid out of its body into an external air bladder. So no matter what the implementation, as long as the water displaced/density can be manipulated, buoyancy control has been achieved.

Chapter 3

Development

Development was always an ongoing process with new version after new version of the computer boards and revision after revision of each of those boards. Development first started back in May 2011 during my summer internship at NAVSEA-PCD, and the work on this project has been on and off due to schedule conflicts, but my development time for this prototype, is nearing its end.

3.1 Design Considerations

There were 5 requirements for the ECS UNIT given by NAVSEA-PCD: 1- Its weight had to be under five pounds maximum, but ideally as close to one to two pounds as possible; 2- Its dimensions must be no bigger than ten inches in height and three inches in diameter; 3- It must be reasonably priced to be expendable; 4- It must have wireless communication; and 5- It had to record the top ten meters of the water column. Additional requirements added to the project from myself were: 6- It should have a 50m depth rating; and 7- It could be reused if retrieved.

The physical design of the ECS Unit's housing was mainly determined by the shape of the AUV payload bay that would hold the ECS Unit. At three inches in diameter, and ten inches in deep bay with a domed ceiling, these dimensions of the AUV bay gave the ECS Unit its size and shape. See figure 3.1 for what the housing looks like when collapsed for storage.

Figure 3.1: ECS unit compressed for storage

Due to the size restraint, the ECS Unit utilizes a telescoping design that allows the bottom of the unit to slide up into the buoyancy chamber. When deployed, the ECS Unit will expand when it inflates the buoyancy chamber to maintain its position at the surface. See figure 3.2 for what the ECS Unit looks like when expanded.

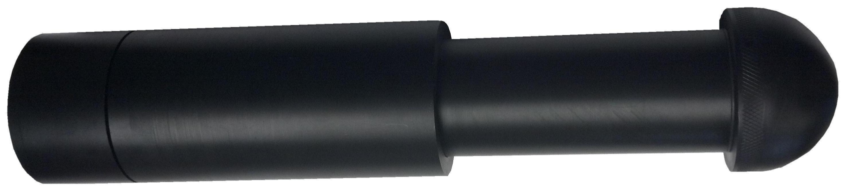


Figure 3.2: ECS unit expanded

The Intended operation of the ECS Unit begins with the unit being deployed either from either an aerial vehicle or AUV into the ocean. To prevent early activation the unit is powered on only when a pair of water contacts become submerged for any longer than a few seconds. When the unit is deployed and becomes activated, it begins its mission as depicted in figure 3.3 on the following page. After the ECS Unit is deployed it releases a limited amount of compressed air to expand the air chamber, and create positive buoyancy. Then, when the unit reaches the ocean surface it establishes a Global Positioning System (GPS) location and makes the initial contact with a Tactical Satellite (TACSAT) (one of 4 in a series of U.S. military experimental reconnaissance and communication satellites).

When it is ready to begin its first dive, the unit opens the buoyancy chambers venting solenoid valve, which allows the air chamber to flood, creating negative buoyancy. While the unit descends to a specified depth it takes readings, and upon reaching that specified depth, it releases the compressed Carbon Di-Oxide (CO_2) gas into air chamber, expelling water, and creating positive buoyancy and ascends to surface. When the unit reaches the surface it gathers new GPS location, and transmits the data to the TACSAT. Then the unit goes into a pre-programmed sleep period until it is time to do its next dive. Upon waking up the unit establishes another GPS location and repeats all the steps over again. Currently the ECS Unit will be able to complete up to 12 dive cycles for a ten meter depth, or only 3 dive cycles for its maximum depth rating of 50 meters.

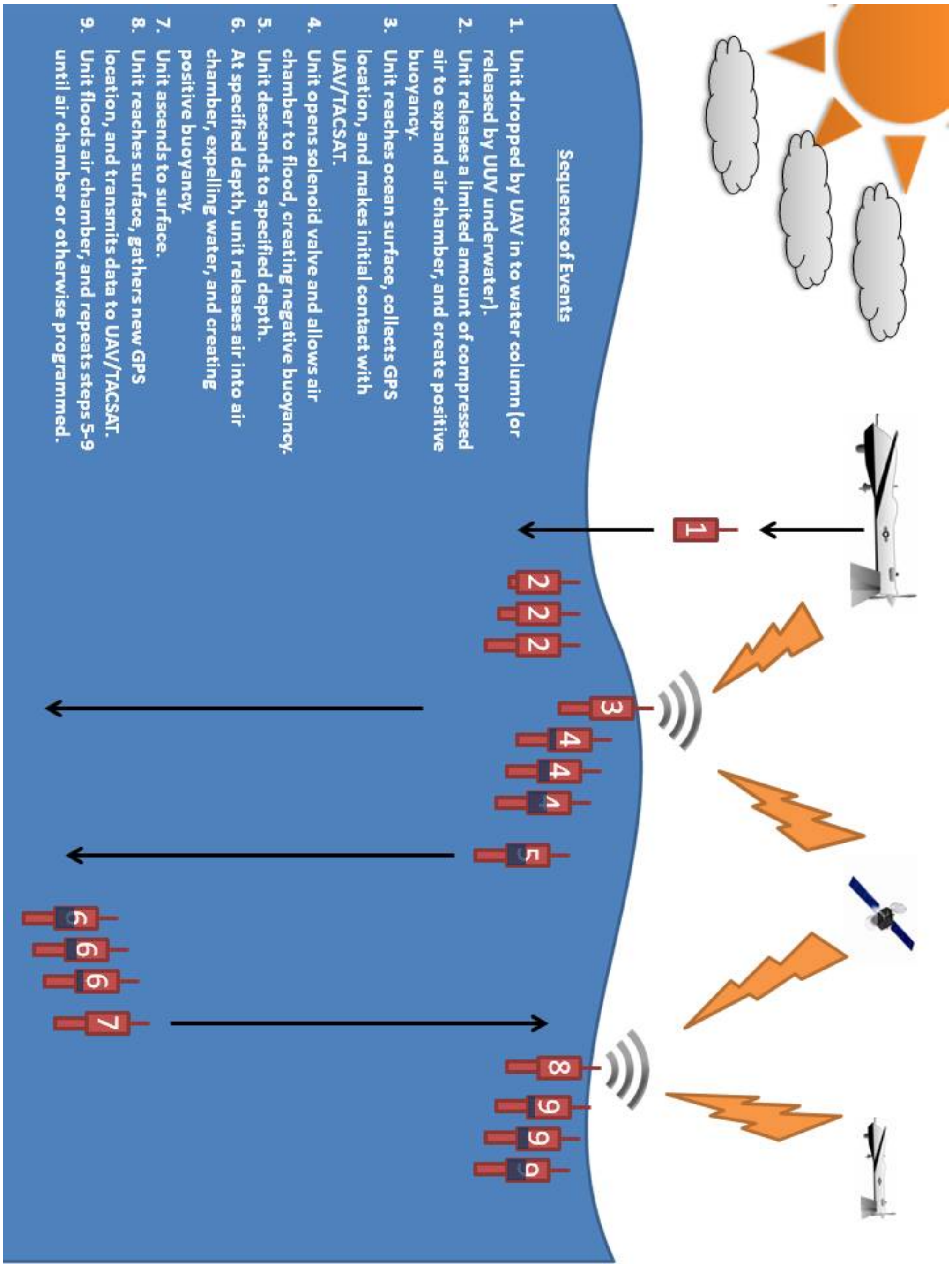


Figure 3.3: ECS Sequence of Events

3.2 Past Prototypes

Each and every prototype and their revisions of the ECS Unit has brought successful solutions to some of the design problems but had also brought into light new problems. The first prototype was constructed to test the proof of concept for the conductivity sensor. From the internship the ECS Unit had originally included a four-pin probe sensor which was later replaced with a two probe design, due to its high complexity and low preciseness. The second prototype was the proof of concept for the buoyancy system that would control its movement in the water column. This prototype was not waterproofed well enough and had a small leak which caused that test to fail. The next prototype version of the ECS Unit was to test the new two probe conductivity sensor after the four-pin probe conductivity sensor was scraped. It is not impossible to get an accurate 4 concentric circle probe to work well, but it would take a great deal of effort and for an expendable sensor it would be too counter productive to the project. The current version (prototype number 4 - revision 2) of the ECS Unit may be even more tuned and perfected then what its current state is, like replacing the custom made pressure regulator with an off the shelf regulator, however more outsourced parts can possibly lead to a higher price tag. Since the ECS Unit is supposed to be expendable there is a fine line on what is an acceptable final cost per unit and what is too costly for an expendable unit to be produced. How the design problems were discovered and overcome, for each subsystem, are discussed in detail in the remaining sections of this chapter.

3.3 Buoyancy System

The buoyancy system for the ECS Unit uses compressed CO_2 gas to displace water in its buoyancy chamber, thus changing the density of instrument platform to make it positively buoyant. To make it negatively buoyant so it can dive, the unit opens a solenoid valve that vents the CO_2 gas in the buoyancy chamber, allowing the chamber to flood with water again. The buoyancy system utilizes one 16g CO_2 canister, one CO_2 puncture device, one pressure regulator, two solenoid valves, and a flood-able chamber. The 16g CO_2 canisters pressure when full is nominally around 900-1000pounds per square inch (psi). The 16g CO_2 canisters pressure posed a problem for the 5 volt miniature solenoid valves that have only a 100psi rating. To remedy this, a pressure regulator is needed. Pressure regulators are used to reduce the pressure acting on downstream components of a pneumatic circuit [11]. For testing purposes it was determined that a pressure regulator built in the FIT's machine should save on money on the prototype to use in other sections of the project. The first version of the buoyancy system was designed with a paintball markers solenoid valve and was designed for an output of 110psi (see figure 3.4). The pressure regulator was built out of half inch square aluminum stock.

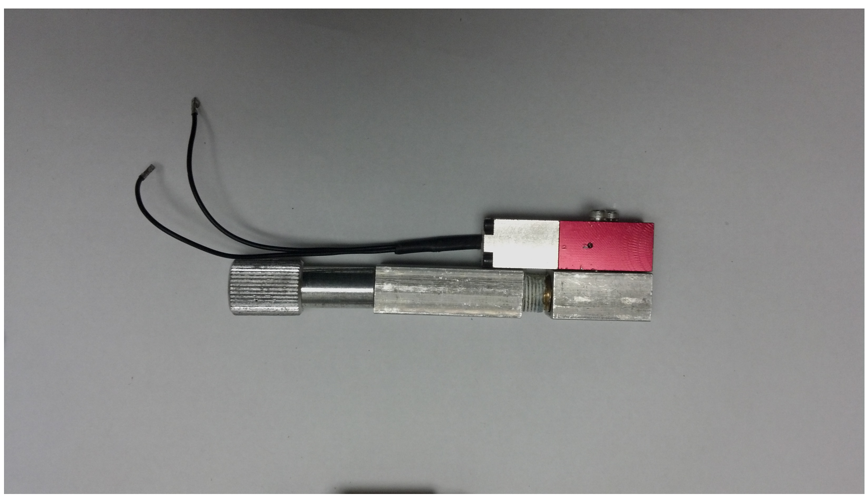


Figure 3.4: Old Regulator

On one input side it was threaded for a Schrader valve similar to what is found in a bicycle tire. This was used to connect the CO_2 puncture device to the regulator. From this opening a 0.106in hole was drilled into the reducing side of the regulator, which had an internal diameter of 0.375in. This was just large enough to fit in the seat of the pressure regulator (0.343in). On top of the seat was a spring with 6 lbs of additional force to the force of the pressure on the reduction side. These measurements were determined by the pressure regulator force balance equation seen below [10].

$$F_{sp} + \frac{P_2}{A_2} = \frac{P_1}{A_1} \tag{3.1}$$

A second version of the buoyancy system was designed to be a bit smaller. A new pressure regulator was needed and designed to accommodate a smaller solenoid valve that maxed out at 100psi (see figure 3.5).

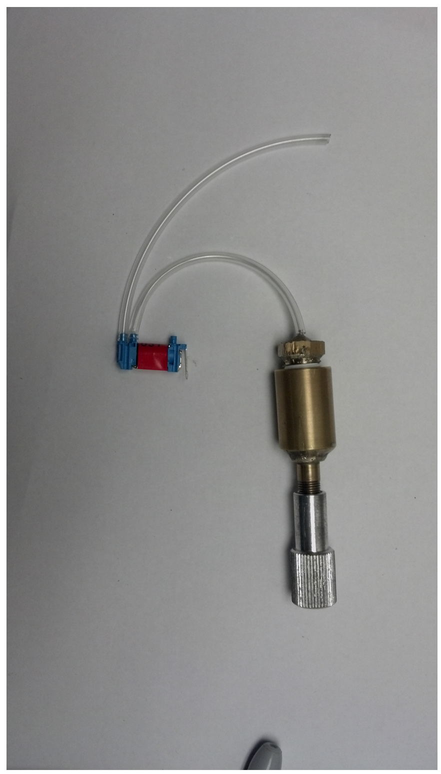


Figure 3.5: New Regulator

This new version of a smaller pressure regulator was made out of 1in round brass stock, and rather then have a female threaded section to attach the CO_2 canister, it included a male threaded section to help shorten the apparatus.

From the input side of the regulator a 0.040in hole connect to the reduction side that had an interior diameter of 0.625in which accommodated a 0.5in seat. With the use of a spring supplying an additional 8 lbs of force a pressure reduction to 80psi was achieved.

While the regulator that was designed worked, it is recommended that a low cost, off the shelf version be used for better quality when the ECS Unit goes into production. The designed pressure regulator that was used was a basic single stage regulator that brought the 1000psi down to 80psi (size limitations prevented a further pressure reduction), but due to the single stage design the pressure regulator leaves about a 100psi remaining in the CO_2 canister. A two stage pressure regulator is recommended for use in the ECS Unit so more of the compressed air may be utilized in a further reduced pressure.

Currently the buoyancy system allows the ECS Unit to be able to complete only up to 12 dive cycles for a ten meter depth, or only 3 dive cycles for its maximum depth rating of 50 m. How those numbers were calculated are shown below using the following equations and data constants from [27].

First take the 16g of compressed CO_2 and convert it to a molecular weight:

$$16gx(1mole/44g) = 0.3636mole \tag{3.2}$$

Then take that weight and calculate its volume (L) at the exhaust ambient pressure and divide it by the amount of volume needed in the buoyancy chamber to make it positively buoyant and subtract 1 cycle for a factor of safety:

For a 10m dive depth exhaust ambient pressure will be 2 atmospheres (atm):

$$V = [(0.3636mole)(0.0822)(293)]/2atm = 4.38L \tag{3.3}$$

$$\frac{4.38L}{0.3369L} = 13 - 1 = 12 \; Dive\ Capability \tag{3.4}$$

For a 50m dive depth exhaust ambient pressure will be 6 atmospheres (atm):

$$V = [(0.3636mole)(0.0822)(293)]/6atm = 1.46L \tag{3.5}$$

$$\frac{1.46L}{0.3369L} = 4 - 1 = 3 \ \textit{Dive Capability} \tag{3.6}$$

3.4 CTD Unit

The proof of concept CTD design, used a liner four-pin probe design. The probes were placed in a line and spaced equally apart (see figure 3.6).

Figure 3.6: Prototype 1

The theory behind the four-pin probe design was to get a more stable conductivity reading from the water while avoiding any electroplating of the probes. In the four-pin probe design, the outermost probes alternate pushing/pulling current through the water, while the two middle probes read the voltage. The two probes reading a voltage at separate locations, gives a change in voltage over the distance between the probes. Combining the change in voltage with the known current being sent across the outermost probes will produce the resistivity of the water, and along with a temperature reading, salinity can be calculated by using the PSS equation (see section 1.3).

Getting this design of the CTD probe to work proved to be harder then thought. The readings were not stable at all, so in the next prototype of the CTD system the 4 pin probe was changed to a linear four-pin probe design with a four concentric circular pin design (see figure 3.7).

Figure 3.7: 4 Pin Concentric Circular Probe

This proved to be a bit more stable but was still fluctuating the readings quite a bit. After doing some more research on the four-pin probe design it was discovered that a lot more calculations were needed then what is normally used to get the resistivity of the water. Since there were multiple probe pins, the current flowing around each pin converted each pin into acting like a magnet, which also changed the cell constant. The cell constant would have to be normalized for the current flow of an electrical dipole field [26]. Due to time constraints this proved to be too much of a hassle and the four-pin probe was discarded for the usual two pin design.

Redesigning the CTD probe yet again improved the readings a great deal, but the between the three different current sensor chips being used, it still proved to be a wild beast to tame. To save on time, the finicky CTD was replaced a new CTD design that incorporated an Atlas conductivity chip. The Atlas proved to be worth its money several times over. It was a small embedded CTD chip that any probe could be hooked up to it as long as it had the same cell constant it was designed for. It required a cell constant of 10 for solutions that have a high concentration of salt like oceanic waters. A new probe was designed to have the cell constant K = 10 (see figure 3.8).

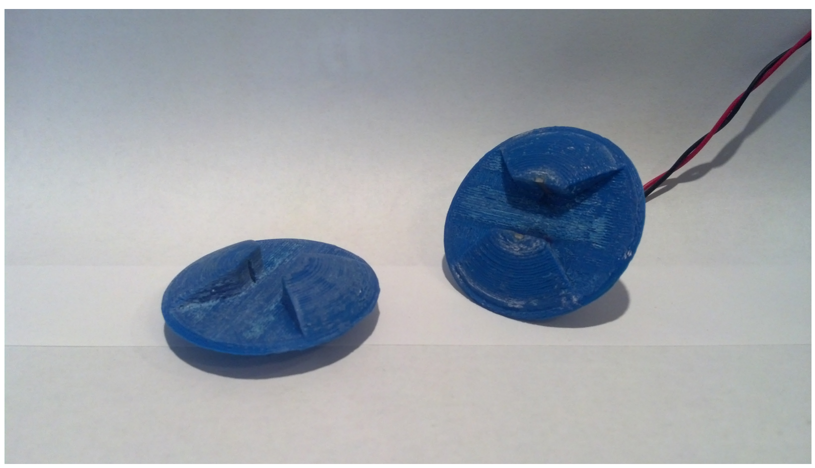


Figure 3.8: New Probe for K=10

This incorporation of a new design for the CTD probe proved to be a tricky process still. First of all the cell constant K = 10 was tricky to set up by hand since it needed to be as precise and accurate as possible. It was first designed with two cylindrically shaped stainless steel pins set at a distance of one centimeter apart, the stainless steel quickly corroded as soon at a current was passed between them. Therefore these pins were upgraded to gold plated nickel pins that were slightly larger, and thus placed about two and a half centimeters apart. These gold plated nickle pins also corroded, albeit at a much slower rate. It was decided then to upgrade to platinum pins, after being tired of the pins corroding and throwing off the readings of the Atlas Conductivity chip. The jump was made to platinum pins, and those have never reacted poorly to current passing between them while submerged in the saltwater environment like the pins made of the lesser grade materials had. Platinum is very expensive so the smallest possible probe was designed that we could physically build and test. As seen above in figure 3.8, the pins were changed to the miniature flat bar shape in the existing probe. Each of these pins are 2 millimeters in width and 6.35 millimeters in height, with a spacing of 12.7 millimeters in-between. This was determined by using the cell constant value of $K = 10$ and working the equation backwards, using the known surface area of the pins, to get the distance between them.

3.5 System Electronics Integration

The ECS Unit is as basic as an autonomous instrument can get. Keeping it basic helps to keep the cost down, and since that was one of the requirements it is pretty important. Making up the ECS Unit are five main electronic systems: GPS, CTD, Communications, Data Logger, and Controller. Each serves a crucial role in the operation of the ECS Unit. The GPS system shows how the unit is drifting through the currents in the ocean, but more importantly shows the location where each dive was performed. The CTD system handles all the water sampling that the unit was made to do, and is one of the most important systems of the ECS Unit. The communications system allows the data to be offloaded of the ECS Unit; but as a backup, there is a micro SD card on board that all the data is saved to. That brings us to the data logging system, without which there would be no data to work with. And lastly but not least, the controller or brains of the unit is a PIC18LF4520 chip.

3.5.1 GPS

GPS or Global Positioning System, was first established back in 1978 by the United States Department of Defense. By sending a signal to a minimum of four satellites, a position can be triangulated on the Earths surface down to about a ten meter radius [23]. GPS operations depend on a very accurate time reference, given by atomic clocks from the U.S. Naval Observatory, and each GPS satellite has atomic clock on board [9]. The ECS Unit uses the Venus GPS board from Sparkfun Electronics (see figure 3.9 below).

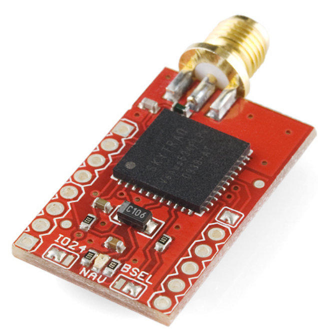


Figure 3.9: Venus GPS: [7]

It outputs the standard National Marine Electronics Association (NMEA) sentences with update rates up to 20Hz. It requires a regulated 3.3V supply to operate, and when powered up uses up to 90mA. The main NMEA string being output, that the ECS Unit looks for from the Venus GPS, is the GPRMC string which provides a valid signal indicator, date, time, latitude, longitude, speed, bearing, and magnetic variation (and happens to be the shortest string the Venus GPS chip outputs). This posed a problem of actually generating more information then what we were looking for. The only data points needed from this string for the ECS Unit are the valid signal indicator, date, time, latitude, and longitude. So a separate GPS parse loop was created. The parse loop looks for the start of the GPRMC string and then saves each character until it sees the end of the string. With the GPRMC string saved in memory, the parsing loop then looks for commas that are used to separate the values of each data point. Using the commas the parse loop separates the data points that is wanted and discards those that are not.

3.5.2 CTD

The CTD is the system that handles the water sampling that the unit was made to do, and it uses the E.C. Circuit from Atlas Scientific (see figure 3.10 below). Atlas Scientific claims that the chip is accurate enough for lab work, yet rugged enough for a long-term field deployment, which is why I chose it. It was almost as simple as plug and play, it required a quick calibration with its included calibration solutions, and it then reads out three values: conductivity, total derived solids (referenced to KCL), and salinity.

Figure 3.10: Atlas E.C.: [24]

The Atlas chip outputs these values in a string, which posed the same problem as the GPS but slightly different. Rather then needing to parse the data being output, the problem was looking for the carriage return character at the end of the string. Unlike most chips outputting strings that end with the standard null character, the Atlas chip ends its strings with carriage returns. So a parse loop was used to save the string and look for when the data ended. The salinity values are derived from the Practical Salinity Scale (see section 1.3). The ECS Unit uses one of the Atlas chip's commands to take a temperature calibrated reading. After sending the ambient temperature value to the Atlas chip it responds by sending back the temperature calibrated reading. This process takes just under a second to do (which is a long time by electronics standards), and this regulates the rate of sampling for the ECS Unit to 1Hz as it descends through the water column.

3.5.3 Communications

The communication system of the ECS Unit allows for its data to be offloaded. The XBee chip from Digi is what the prototype uses currently (see figure 3.11). It uses a form of Wireless Fidelity (WIFI) called ZigBee Protocol, and it gives a data transfer range of about one hundred and twenty meters line of sight, which was the problem with this communications system. To retrieve any data the receiver has to be located fairly close as the data transfer range is rather optimistic. Like the Venus GPS it is powered with 3.3V, and when transmitting data uses about 200mA.

Figure 3.11: Xbee WIFI: [8]

When the ECS Unit goes into production, the XBee chip will be replaced with an Iridium Satellite Communications Chip (see figure 3.12). This chip will be able to communicate with the NAVYs TACSAT network providing data access from the ECS Unit anywhere in the world.

Figure 3.12: Iridium: [3]

3.5.4 Data Logging

The data logging system is run by OpenLog which is an open source data logger (see figure 3.13). This logger takes any serial stream that is sent to it and will log it. The only problem that arose with this system was actually from a limitation of the controller.

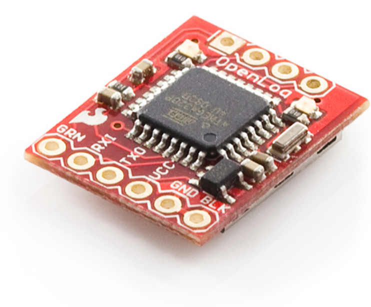


Figure 3.13: OpenLog: [5]

A secondary software data transition/receiving port was set up to fix it. The OpenLog pretty much logs continually, but commands can be sent to it to have other operations done, for example: sending it the command "*new File*" will create a new file named File and then sending the command, "*write*" puts it back into logging mode again, but now, in doing so, saves data to the new file. Cycling the power creates a new file named with incrementing numbers. The ease of use from the OpenLog made it an easy choice to use for the data logging operations for the ECS Unit.

3.5.5 Controller

The controller system or the brains of the ECS Unit uses a PIC18LF4520 chip from Microchip (see figure 3.14) to control the units functions. PIC microcontrollers can be found in many kinds of applications like smoke detectors, battery chargers, LED flashlights, and advanced medical devices. Microchips PIC 18LF4520 is an 8-bit microcontroller that utilizes a 16-bit program word architecture [21]. The PIC18 family is one of the most popular chip families for embedded systems that support both 3V and 5V applications like the ECS Unit. The LF version of the chip has a power greater supply range of 2V to 5.5V and draws a minimal current when compared to other chip families. In this application it controls when the GPS, WIFI, CTD, and data logger are switched on and off, parses the GPS and Atlas Chip output strings, writes/reads data to and from the data logger, and sends the data out through the WIFI chip. Not only does it control and interact with all the systems, but also controls itself, meaning the it puts itself to sleep in between dives and wakes up when it is time to start a new dive. There were two main problems that came to light when working with the PIC18LF4520, and that was a memory problem and it also had only a single hardware data transfer port. The memory problem that was discovered was when the ECS Unit was saving all the dive data to its on board flash memory. There was just too much data to store on each dive before transferring it to the external memory of the OpenLog, and the PIC18LF4520 would start to become glitchy. This was solved just by saving the data to the data logger as it was sampled. The other problem that was known but thought could be worked around was the fact that this chip only had one hardware data transfer port. The GPS, WIFI, CTD, and Data Logging systems all use this port and sometimes at the same time. To work around this a secondary software data transfer port was set up and fixed that problem. The PIC18LF45290 is most definitely an integral part of the ECS Unit.

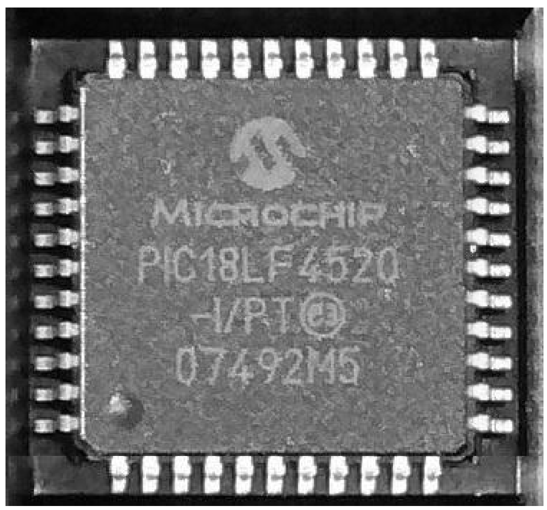


Figure 3.14: Pic chip: [21]

3.6 Software and Application

When the ECS Unit is activated, it runs a start-up sequence, then moves into the main operations loop, and exits only when it reaches its maximum number of pre-programmed dives. This sequence of events is depicted visually in the flow diagram in figure 3.15. Upon activation the ECS Unit determines whether it floating at the surface of the water or not by sampling the pressure sensor, and GPS for a valid signal. If it is not located at the surface it inflates the buoyancy chamber and rises to the surface. Once it has determined that indeed its location is at the surface of the ocean, the ECS Unit then makes initial contact with the satellite and records its location with a GPS reading, parses the GPS signal and records the date, time, and location. Once it has recorded that information, the ECS Unit starts up the dive sequence. The dive sequence consists of shutting down the communications chip and GPS chip, starting the continual background sampling of the pressure sensor, as well as venting the buoyancy chamber, allowing it to flood to start the dive. While the ECS Unit sinks, it runs a time based sampling loop until the unit reaches the pre-programmed depth for its dive.

The time based sampling loop takes a sample of the water column by reading the water temperature, sending that value to the Atlas conductivity chip (which spits back a temperature compensated reading in the form of conductivity, TDS, and salinity), and reading the pressure to calculate depth. It then takes the temperature, depth, conductivity, TDS, and salinity readings and sends that data out to the data logger to be recorded every second. Upon reaching the pre-programmed depth it exits the sampling loop and begins the end dive sequence. This turns off all the sensors, minus the pressure sensor, and sends a set of rapid bursts of the compressed CO_2 into the buoyancy chamber to make it positively buoyant so it can rise back up to the surface. The ECS Unit continues to sample the pressure sensor to see if it is still sinking, and if it senses that there was not enough CO_2 released into the buoyancy chamber to reverse its direction, it would then send another set of rapid bursts of CO_2 into the buoyancy chamber. When the ECS Unit reaches the surface, the GPS is turned back on and the ECS Unit waits until it receives a valid signal and then records is location again. At this point the GPS is turn off and the communications chip is activated. The ECS Unit then sends out the data packet which consists of its initial location, dive data, and resurfaced location. After all the data from that dive gets sent out, the ECS Unit increments the dive count number (initialized at zero) by one. The dive count number is then checked against the max number it was programmed to do. If the ECS Unit has not reached that number the unit then powers down into a low power state and sleeps for a pre-programmed amount of time before its next dive. If the max number of dives was reached, the unit then opens the buoyancy chambers venting valve, powers off, and sinks beneath the surface of the ocean for the last time. (If the navy would like the unit to be retrievable, rather than powering off and sinking to the ocean floor, it can chirp its location every few hours so it can be found and get picked up.)

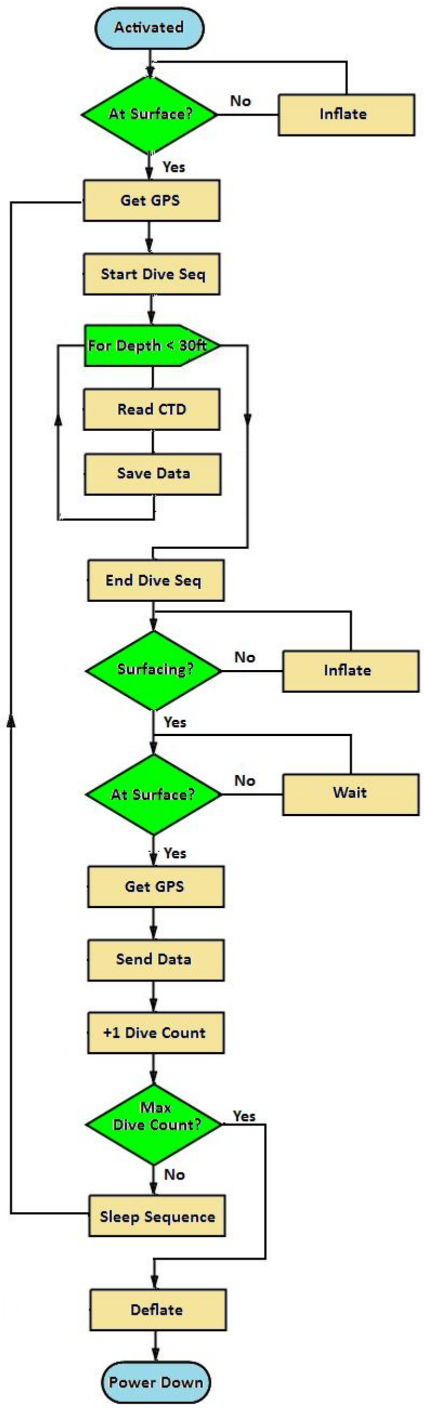


Figure 3.15: ECS unit Program Flow Chart

Chapter 4

Testing and Analysis

Each of the subsystems of the ECS Unit were tested individually, the most important and recent one, being the CTD system was tested in this study. During the design and construction phase for each of the past prototypes, the ECS Unit was bench tested for functionality. This happened sequentially as each subsystem was adding into the project. The GPS, Communications, and Buoyancy systems were all tested satisfactorily in the previous prototypes. This most recent prototype of the ECS Unit tested the CTD system against YSI's YSI-85 (a hand-held instrument that measures oxygen, conductivity, salinity, and temperature) and IDRONAUT's Ocean Seven 320 Plus (a standard CTD used in ship casting operations). At FIT it is used in field studies and in-lab use. The IDRONAUT Ocean the Seven 320 Plus is a full scale CTD that can be mounted in a rosette wheel, which is what the configuration of the instrument is that FIT uses in the Marine Field Project summer cruises. The test plan for this new prototype of the ECS Unit briefly re-investigated the buoyancy, GPS, Communications, and Control system again; but the test plan mainly focused on the CTD system. The test plan for the ECS Unit consisted of three phases. The first phase of the test plan started with a basic bench test of all the subsystems. Then the second phase covered the characterization bench test for the CTD system. Finally, the third phase encompassed a controlled field test of the CTD system. Upon completion of the testing phase, analysis was done comparing the readings of the ECS Unit to the readings of a YSI-85 and IDRONAUT-CTD, and then comparing the functionality of the ECS Unit to the XCTD.

4.1 Bench Test

The bench test of this most recent prototype of the ECS Unit consisted of a brief re-test of the prior completed subsystems, and an initial bench test of the CTD system along with a second bench test for the characterization of the CTD system readings. The electronics and probes of the prototype were tested in a temporary housing. This was done to save time and effort while testing the CTD system. The housing was that of a water proof case used for boating purposes to keep valuable electronics, like cell phones and cameras, dry (see photo 4.1 below).

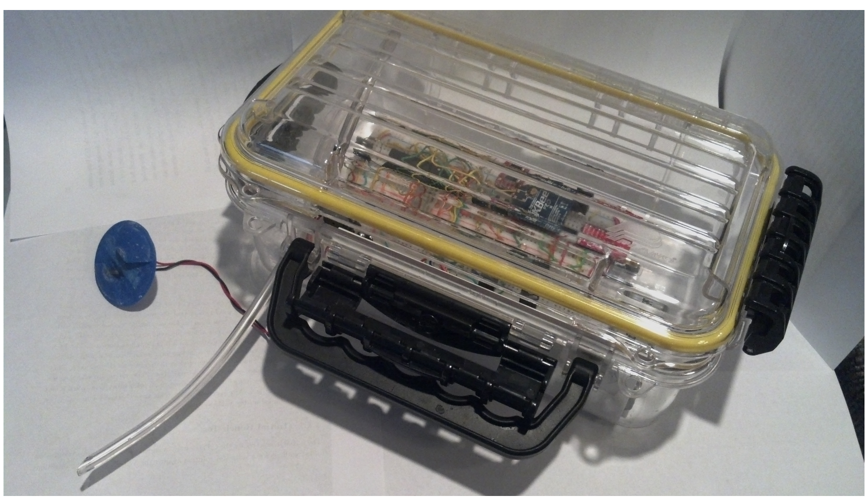


Figure 4.1: Dry Box Housing Electronics Bread Board

The reason there was a brief re-test of the GPS, Buoyancy, Communication, and Control systems was to check that the functionality of each of them transferred over to this prototype from the last version. As anyone working with embedded electronics will find out, things just tend to go wrong when you transfer electronics from system to system, even if nothing has changed. The ECS Unit was no exception; subsystems would go haywire on their own overnight (especially just after combining a new or old subsystem with the project). They were then looked into, debugged, and fixed. The CTD system was not exempt from this problem either, which is why a brief re-test was done on all prior subsystems when it was added to the project. After each of the systems were checked for functionality, testing continued for the CTD system.

4.1.1 Initial Bench Test

The Initial bench test for the CTD system included calibration and a verification of its readings for a saltwater solution against the YSI-85. Using the directions given in Atlas Scientifics Conductivity Circuit Data sheet [24], the ECS Units custom made probe was calibrated. To calibrate the sensor, the following steps were completed in order. After connecting to the chip the probe type was set. Setting the probe type was done by sending the command P,3$< cr >$ to the chip ($< cr >$ simply denotes a carriage return must be sent at the end of the command) for a probe with a K = 10 value like the probe used in the ECS Unit. After setting the probe type, the sensor was to be calibrated for a dry probe reading, this was done by entering Z0$< cr >$. After calibrating for the dry probe, the high side calibration came next followed by the low side calibration using the calibration solution provided by Atlas Scientific (See figure 4.2 below).

Setting the high side calibration started with placing the probe into the $\mu S'$ 90,000 solution. When the probe was immersed in the high end solution the next step was to put the atlas chip into a continuous reading mode by sending the command C$< cr >$. After waiting about five minutes (just to make sure the readings were stable), the high end calibration command was sent, Z90$< cr >$. With the high end set, the next step was to set the low end calibration. This was done by placing the probe into the $\mu S'$ 62,000 solution, setting the chip to run in continuous mode again, and waited another five minutes. After waiting,

Figure 4.2: Calibration Solution

the low end calibration command was sent, Z62$< cr >$, and the calibration was completed. A final check was done to test of the calibration of the salinity reading of the ECS Unit, which was verification against the YSI-85. It was seen that for a salinity solution of 26 Parts Per Thousand (ppt), the ECS Unit read 3ppt higher than that of the YSI-85, which is a key note for the testing and analysis section.

4.1.2 Characterization Bench Test

Upon being calibrated the ECS Unit readings were characterized, meaning the readings were tested in varying salinity and temperatures to see how or if their values we affected by changing those variables. This test was fairly simple as well, it consisted of testing water with three different salinity at the same temperature, then testing one of the salt water solutions at different temperatures. These solutions were set in separate pales that could fit all three instruments at once. The setup for this test is shown in figure 4.3 below.

Figure 4.3: Bench Test Set Up

The first step in doing this test was to acquire some saltwater solutions to test. Ten gallons of saltwater and five gallons of reversed osmosis filtered water were bought from the local aquarium store, and the saltwater was tested by a salinometer to have a salinity of 26ppt. This was then divided up into three separate solutions one of which the salinity was changed by diluting the solution with the reversed osmosis water to a salinity of 18ppt. Then the reversed osmosis filtered water and the three saltwater solutions were stored in an air conditioned room to bring their temperature down to simulate normal ocean temperatures.

The first part of the characterization test was to measure the salinity of different saltwater solutions while all at the same temperature ($22°C$). The first solution measured was the fresh water. This was done with all three instruments at the same time. The IDRONAUT-CTD was placed into the freshwater first since it was the largest of the three instruments, then the probes from the YSI-85 and the ECS Unit were placed inside the pales. After waiting for a few minutes to allow their readings to stabilize, their readings were recorded by hand. They all read an average reading of about 0ppt which is a good reading for fresh water. This procedure was then repeated for the 18ppt and the 26ppt solutions (see table 4.1).

Table 4.1: Characterization Test 1

Steady Temperature ($20°C$) $-$ *Variable Salinity*

Solution	IDRONAUT CTD	YSI-85	ECS Unit
RO (0ppt)	0.042	0.1	3
18ppt	18.128	18.5	21
26ppt	26.174	26.7	29

For the reverse osmosis water sample, the instrument readings were as follows: the IDRONAUT-CTD read 0.042ppt, the YSI-85 read 0.1ppt, and the ECS Unit read 0.0ppt. For the 18ppt saltwater solution the IDRONAUT-CTD read 18.128ppt, the YSI-85 read 18.5ppt, and the ECS Unit read 21ppt. For the 26ppt saltwater solution the IDRONAUT-CTD read 26.174ppt, the YSI-85 read 26.7ppt, and the ECS Unit read 29ppt. (Note: that the ECS Unit consistently reads 3ppt higher than the other CTDs. This is what will be used to calculate the unity of the CTD sensor.)

The second part of the characterization test was to measuring the salinity of a saltwater solution (26ppt), at different temperatures. The solution was first measured at $19.5°C$. Just like in the previous characterization test this was done with all three instruments at the same time. The IDRONAUT-CTD was placed into the saltwater solution first and then the probes from the YSI-85 and the ECS Unit were placed inside the pales. After waiting for a few minutes to allow their readings to stabilize, their readings were recorded by hand. The YSI-85 and the IDRONAUT-CTD read an average reading of about 26ppt which was expected and the ECS Unit read 29ppt. This procedure was then repeated for at the temperature of $22.1°C$ and $28.2°C$ for the 26ppt solution. As expected the salinity readings did not change drastically, only the temperature (see table 4.2).

Table 4.2: Characterization Test 2A

Steady Salinity (26ppt) − Variable Temperature			
Solution	IDRONAUT CTD	YSI-85	ECS Unit
19.5°C	26.173	26.6	29
22.1°C	26.175	26.8	29
28.2°C	26.176	26.7	29

Then the remaining 26ppt saltwater solution that was kept in the air conditioned room was used to show the dynamics of the water while raising the temperature from 19°C to 28°C. This was done to see how the readings would change when passing through a thermo cline, and since we had no readily available way of actively bringing the temperature down quickly, the temperature was brought up quickly. This acted just like passing the CTD probes through a reverse thermo cline in a column of water. In this test the ECS Unit was compared to the IDRONAUT-CTD only since the YSI-85 was not set up to record data automatically. As expected again, the salinity readings did not change dynamically, only the temperature changed dynamically (please see figure 4.4 below for values).

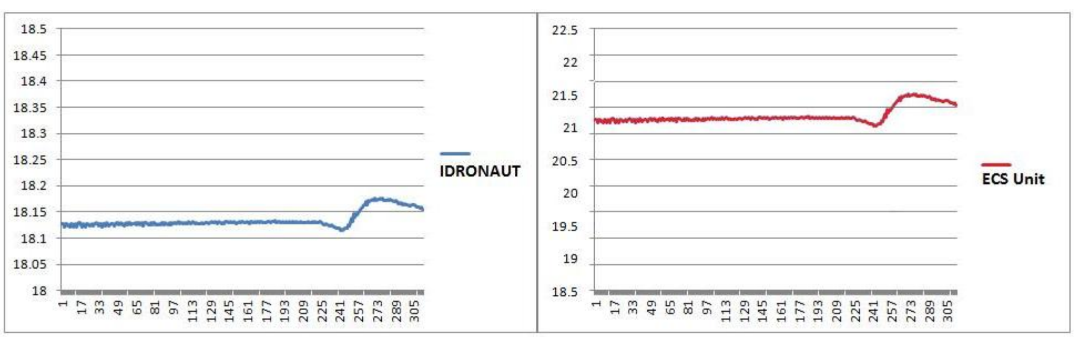


Figure 4.4: Characterization Test 2B

4.2 Controlled Field Test

After successfully passing the characterization testing, the ECS Unit then underwent a controlled field test. This was done at FITs anchorage, located at the mouth of Crane Creek (see location below in figure 4.5).

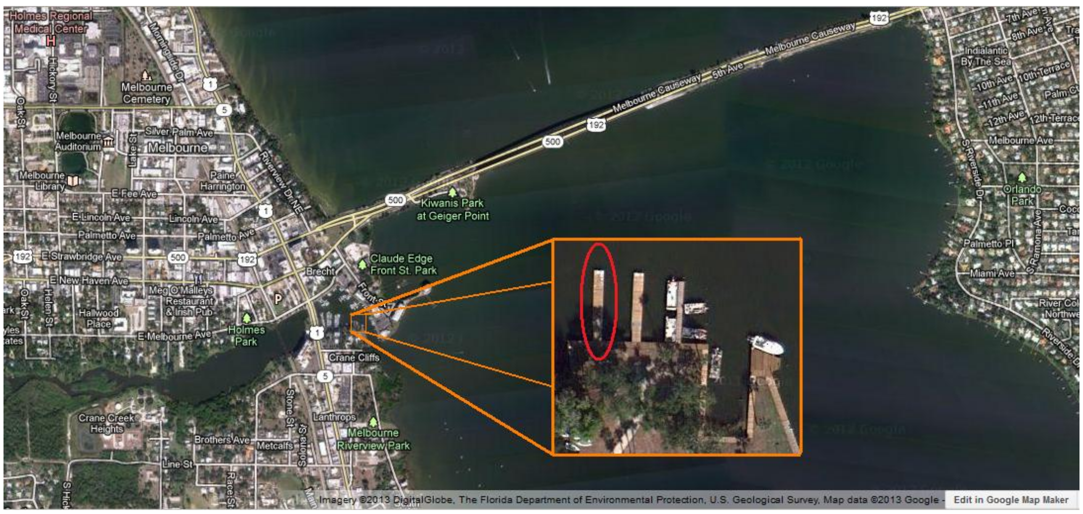


Figure 4.5: Google Maps: FIT Anchorage

In the above photo, off the end of the dock (circled), is where the ECS Unit was tested against the IDRONAUT-CTD. The procedure for this field test was to perform three separate CTD casts off the end of the dock, and compare all three individually and as an average between the two CTD instruments. For each cast the two CTDs were lowered at the same time and rate to the bottom of the marina (which was just under five feet deep), remaining down there for a few seconds to allow a stable reading, then brought back up together (please see figure 4.8 for the graph of the first cast). After the cast, the logged data was recorded and saved in separate files for each instrument, then the process was again repeated until all three casts were completed and their information gathered and logged.

4.2.1 Casts and Averaged Data

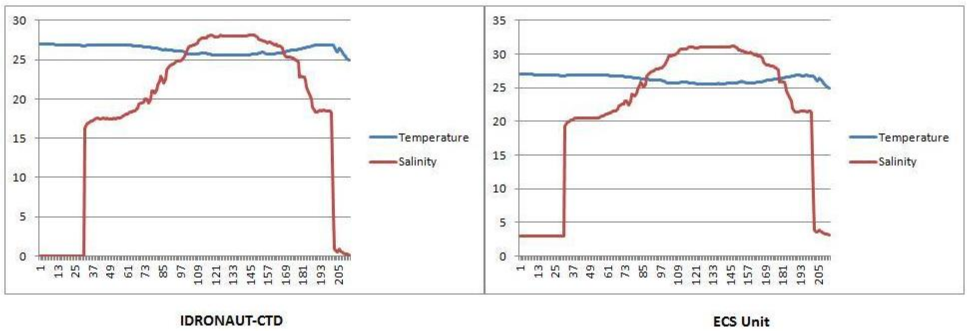


Figure 4.6: Field Test Cast 1

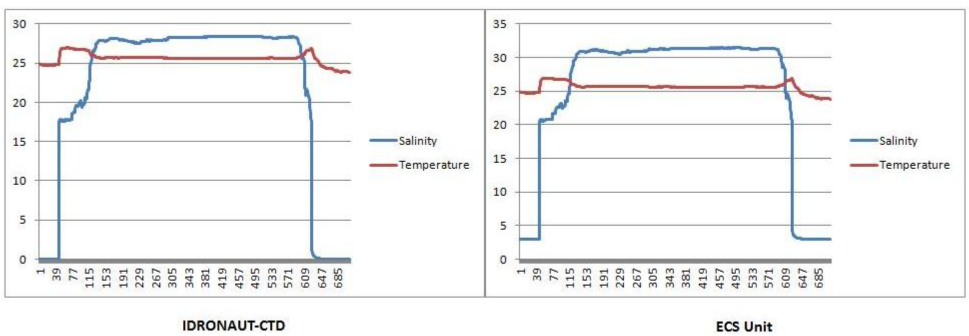


Figure 4.7: Field Test Cast 2

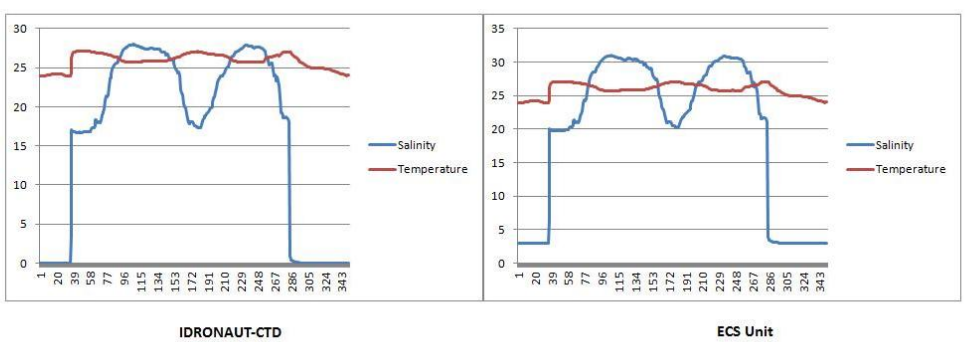


Figure 4.8: Field Test Cast 3

4.3 Analysis of Test Data

The analysis of the test data gathered from all four tests performed on the ECS Unit was compared to the data collected from the IDRONAUT as well as the YSI-85. (NOTE: There is a chance that since the instruments were last calibrated, a % error in the instrument readings could have been introduced, but the focus of this analysis test was to look at the trend of the data. The % error in readings is easily fixable while an error in the data trend is a fundamental problem.) It was noted back in the initial bench test that the ECS Unit had a consistent reading of 3ppt higher than the YSI-85, and this was true of all the tests. The initial bench test, and first part of the characterization tests, were not able to be graphed due to the data not being continuously logged but only recorded by hand, and this was also the case with the first half of the second part of the characterization tests as well. However, the second half of the last phase of the characterization testing was recorded continuously, and therefore this is where the graphical comparison begins for the ECS Unit. The data comparison was made between the instruments by observing and comparing their outputs through the graphical representation of their data.

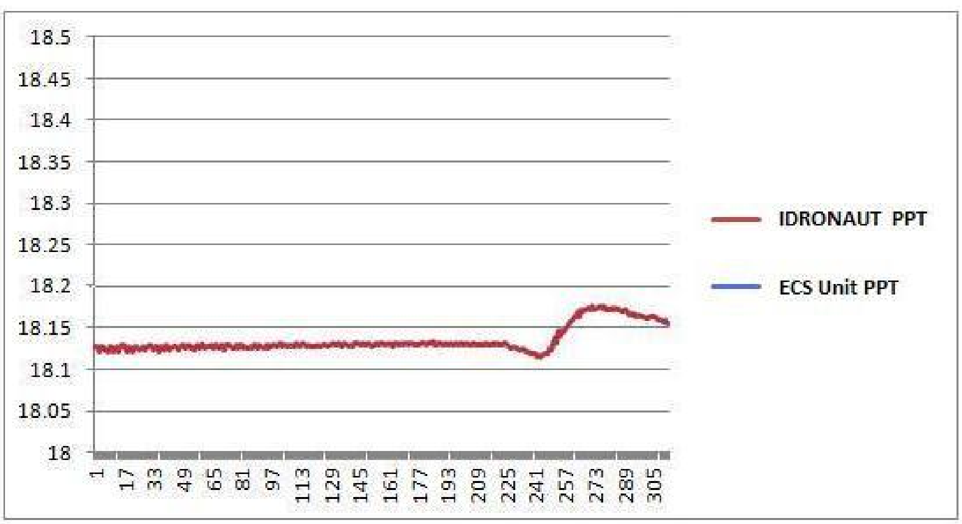


Figure 4.9: Normalized Characterization Test 2B

Looking at the graph in figure 4.9, the trend between the ECS Unit and the IDRONAUT-CTD is extremely similar (when not normalized the consistent offset can be seen and can be easily corrected in the future). As seen in this figure, when the trends are overlaid onto each other, there is no discernible difference between the two outputs. The field test data shows that, between the IDRONAUT-CTD and the ECS Unit, the output readings still have a similar trend with the 3ppt offset. The three separate casts performed at the FIT anchorage were averaged (to normalize the readings) and then graphed. As with the characterization test, the normalized data from the field test showed that the trend between the two instruments was practically identical, with the only difference being the 3ppt offset (see figure 4.10 below).

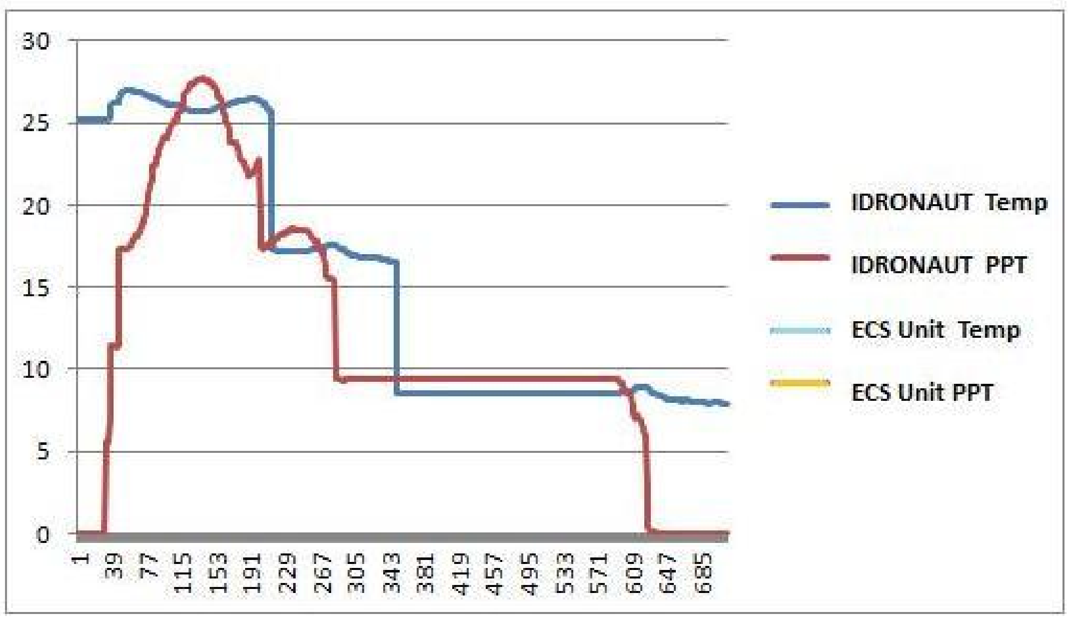


Figure 4.10: Normalized Field Test Casting

When looking at the graph of the normalized field test data, the temperature and conductivity appear to have a stair like trend, this is due to the field test data being split into three sections. The first section covered the lowering of the instruments through the water column, second section was the bottom time of the instruments, and the third section was the retrieval of the instruments. The data files were separated into those three sections to keep the difference in run times, for each field cast, the from skewing the normalized data.

4.4 Instrument Comparison

This comparison study of the ECS Unit sensor reviews the differences between it and the XCTD/AXCTD for cost, life expectancy, depth rating, and utilization. The cost was kept as low as possible for the ECS Unit, and the total price of just parts (including the iridium satellite chip) comes out to be \$982.00. While the price per unit for the XCTD/AXCTD are respectively \$548.50 and \$1,678.32.

The life expectancy of the ECS Unit will be determined by its limiting factor, which is both CO_2 volume and battery life. These two factors are both affected by the frequency of its dives, the depth of its dives, and the duration of hibernation between cycles. For ten meter dive depths, with no hibernation periods, utilizing the maximum number of dives, will place its life expectancy within a 36 hour range. However if hibernation is included in the operation of the ECS Unit, with the maximum number of dives for a ten meter depth, can have an operational range of up to thirty (30) days. When the ECS Unit is programmed to take 50m depth dives and repeating its diving with no hibernation periods, it will place its life expectancy (for the maximum number of 3 dives for that depth) at approximately 18 hours. However though, if hibernation is included, its life expectancy (with the maximum of 3 dives for the 50m depth) spans just under 2 weeks. These numbers are only for is maximum and minimum operating parameters, variations can be made to alter its life span with mixing dive depths and hibernation periods. While the life expectancy for the ECS Unit is variable, the XCTD/AXCTD can be only used for a single dive.

To record the data from the ECS Unit a WIFI chip and inexpensive dongle (a \$45 USB plug-in antenna to receive the signal) can be used to record the data on any laptop or desktop; or through the use of an Iridium satellite communications chip, the ECS Unit can send data to any server in the world for the price of

Iridium's standard messaging rates. The XCTD instrument system requires the data to be transferred through its launching apparatus into its specialized data acquisition system. The cost of the launching apparatus is $1,256.04 and the cost for its specialized data acquisition system is $8,638.47, for a grand total of $9,894.51 worth of components just to retrieve the data. Whereas the AXCTD requires its data to be sent wirelessly to a receiver/data acquisition system which costs $37,131.73. Therefore (as seen in Table in figure 4.13 below) the ECS Unit, even with the shortest deployment configuration, is still more economical than using multiple units of the XCTD/AXCTD along with the man hours and ship time to perform the same mission.

Table 4.3: Instrument Comparison

	ECS Unit	XCTD	AXCTD
Cost per Unit	$982.00	$548.50	$1,678.32
Life Expectancy	3-12 Dives (within a period of 30 days)	1 Dive	1 Dive
Depth Rating	50m	1000m	1000m
Deployment	Single or Multi-use	Single Use	Single Use
DAQ Cost	$45 - WIFI Std Msg Rates - Iridium	$9,894.51	$37,131.75

Currently there are only two downsides of ECS Unit when compared to the XCTD/AXCTD. That is firstly the ECS Unit has a max depth of only fifty meters, whereas the XCTD/AXCTD can go down to one thousand meters, and secondly the ECS Unit only has an accuracy of 1ppt for its salinity readings, whereas the XCTD/AXCTD is 0.01ppt. With some additional work though, the ECS Unit can get down to an accuracy of 0.1ppt, which should be within an acceptable range for the scientific community.

Chapter 5

Conclusion

The ECS Unit was constructed to be a multi-use, expendable instrument for less then \$982.00. It was tested against IDRONAUT's Ocean Seven 320 Plus CTD and YSI's YSI-85 hand-held conductivity meter to prove its functionality. This thesis covered the design, development, and testing of the ECS Unit but more specifically it's conductivity sensor. Between all the system prototypes, the ECS Unit meets all the criteria set forth by NAVSEA-PCD when they approached FIT with these five (5) specific specifications for this instrument. Those requirements were: 1) Its weight had to be under five pounds maximum - the ECS Unit currently weighs 3.8lbs; 2) Its dimensions must be no bigger than ten (10) inches in height and three (3) inches in diameter - Its dimensions are 3in in diameter and 10in in height; 3) It must be reasonably priced to be expendable - It cost \$982.00 to build; 4) It must have wireless communication - It Currently uses WIFI and will be switched to Satellite Communications at a later date; and 5) It only has to record the top ten (10) meters of the water column - It can record up to the top 50m of the water column. Plus the ECS Unit was designed not only to have a maximum dive depth of 50m, but it has the option to be expendable or be retrieved and deployed again if needed. The ECS Unit with its multi-dive capability, low cost, and multi-use option makes it one of the most a viable expendable CTD instruments of today.

5.1 Future Work

Since the prototypes have only been using a WIFI signal, it is required to integrate and test the iridium satellite communications chip into the ECS Unit. Then if needed for future use, additional work can be done on the ECS Unit to adjust its resolution. Since the ECS Unit's accuracy is within 1ppt, fine tuning its resolution could be done to better convert the conductivity readings into a more accurate salinity (within an accuracy of 0.1ppt).

References

[1] Deployment of a ctd. http://www.epa.gov/boldkids/images/ctd-deployment.jpg. Accessed: 2012-09-28.

[2] How argo floats work. http://www.argo.ucsd.edu/float_design.html. Accessed: 2013-1-12.

[3] Iridium satellite communications chip. http://www.iridium.com/Assets/iframes/iridium-9602-slidedeck/images/9602-360/9602-0001.png. Accessed: 2012-10-12.

[4] Lockheed martin sippican expendable probes. http://www.sippican.com/contentmgr/showdetails.php/id/312. Accessed: 2012-10-03.

[5] Openlog data logging chip. http://www.abra-electronics.com/product_images/c/561/DEV-09530__42469_zoom.jpg. Accessed: 2013-2-24.

[6] Subsurface moring. http://www.mbari.org/mars/images/aloha_poster_xl.jpg. Accessed: 2012-10-12.

[7] Venus gps chip. https://dlnmh9ip6v2uc.cloudfront.net/images/products/1/1/0/5/8/11058-04.jpg. Accessed: 2012-10-12.

[8] Xbee wifi chip. http://embedsoftdev.com/wp-content/uploads/2010/10/1261549060.jpg. Accessed: 2013-1-12.

[9] How does gps work? Smithsonian Institution. http://airandspace.si.edu/gps/work.html, 1998.

[10] Pneumatics. In *Process Automation Handbook*, pages 23–26. Springer London, 2007.

[11] Pressure control valves. In *Pneumatic Drives*, pages 193–214. Springer Berlin Heidelberg, 2007.

[12] Dr. Stephen Wood (Chapter Author) Alexander Inzartev (Editor). Underwater vehicles. pages 505–530. In-Tech, 2009.

[13] Sea-Bird Electronics. Ctd profiling instruments. `http://www.seabird.com/products/profilers.htm`, 2013.

[14] Woods Hole Oceanographic Institute. Ocean instruments: Moored profiler. `http://www.whoi.edu/instruments/viewInstrument.do?id=10978`, 2013.

[15] Woods Hole Oceanographic Institution. Conductivity, temperature, depth (ctd) sensors. `http://www.whoi.edu/page.do?cid=1003&pid=8415&tid=282`, 2013.

[16] Jenway. *Conductivity Meters*. Bibby Scientific Limited, 2013.

[17] Joan A. Kleypas, John W. McManus, and Lambert A. B. Meez. Environmental limits to coral reef development: Where do we draw the line? *American Zoologist*, 39(1):pp. 146–159, 1999.

[18] Gary Lagerloef. Ocean motion. `http://oceanmotion.org/html/research/lagerloef.htm`, 2013.

[19] Edward Lewis. The practical salinity scale 1978 and its antecedents. *IEEE Journal of Oceanic Engeering*, 1(OE-5):3–8, 1980.

[20] Herman Medwin. Sounds in the sea from ocean acoustics to acoustical oceanography. pages 1–18. Cambridge University Press, 2005.

[21] Microchip. Microchip 8 bit pic microcontrollers. `http://www.microchip.com/\downloads/en/DeviceDoc/39630h.pdf`, 2012.

[22] NASA. Salinity. `http://science.nasa.gov/earth-science/oceanography/physical-ocean/salinity/`, 2013.

[23] Robert Pagliari. *Hybrid Renewable Engergy Systems for a Dynamically Positioned Buoy.* Master Thesis, Florida Institute of Technology, 2012.

[24] Atlas Scientific. Conductivity circuit datasheet. `http://oceanmotion.org/html/research/lagerloef.htm`. Version 3.0 DataSheet was accessed: 2013-1-28.

[25] San Diego University of California. Argo - part of the integrated global observation stratagy. `http://www.argo.ucsd.edu/index.html`, 2013.

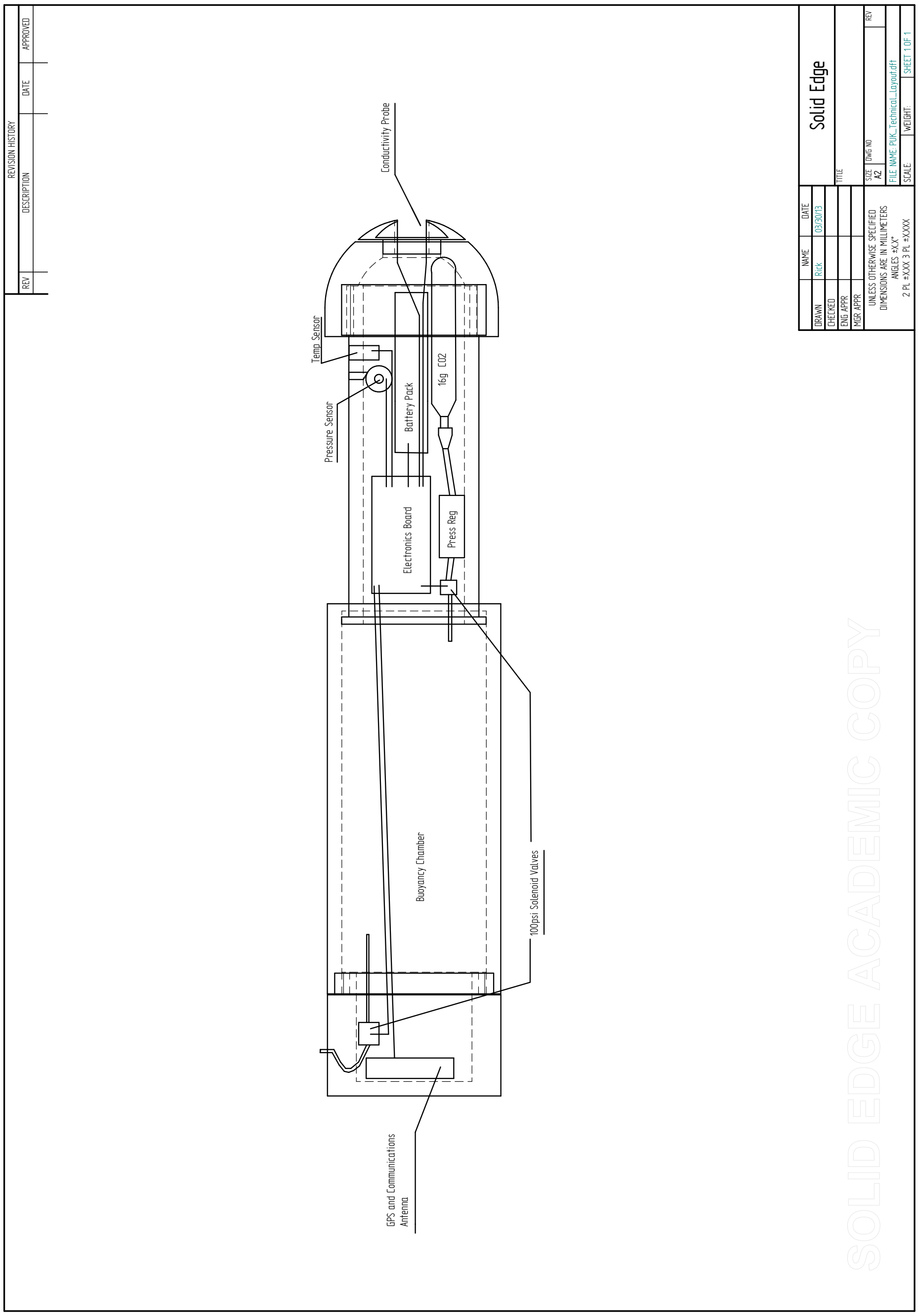
[26] P. Bergveld W. Olthuis, W. Streekstra. Theoretical and experimental determination of cell constants of planar-interdigitated electrolyte conductivity sensors. *Sensors and Actuators*, 24(25):pp. 252–256, 1995.

[27] P.J. Linstrom W.G. Mallard. Carbon dioxide, nist standard reference database 69: Nist chemistry webbook. `http://webbook.nist.gov/cgi/cbook.cgi?ID=C124389&Mask=1`, 2011.

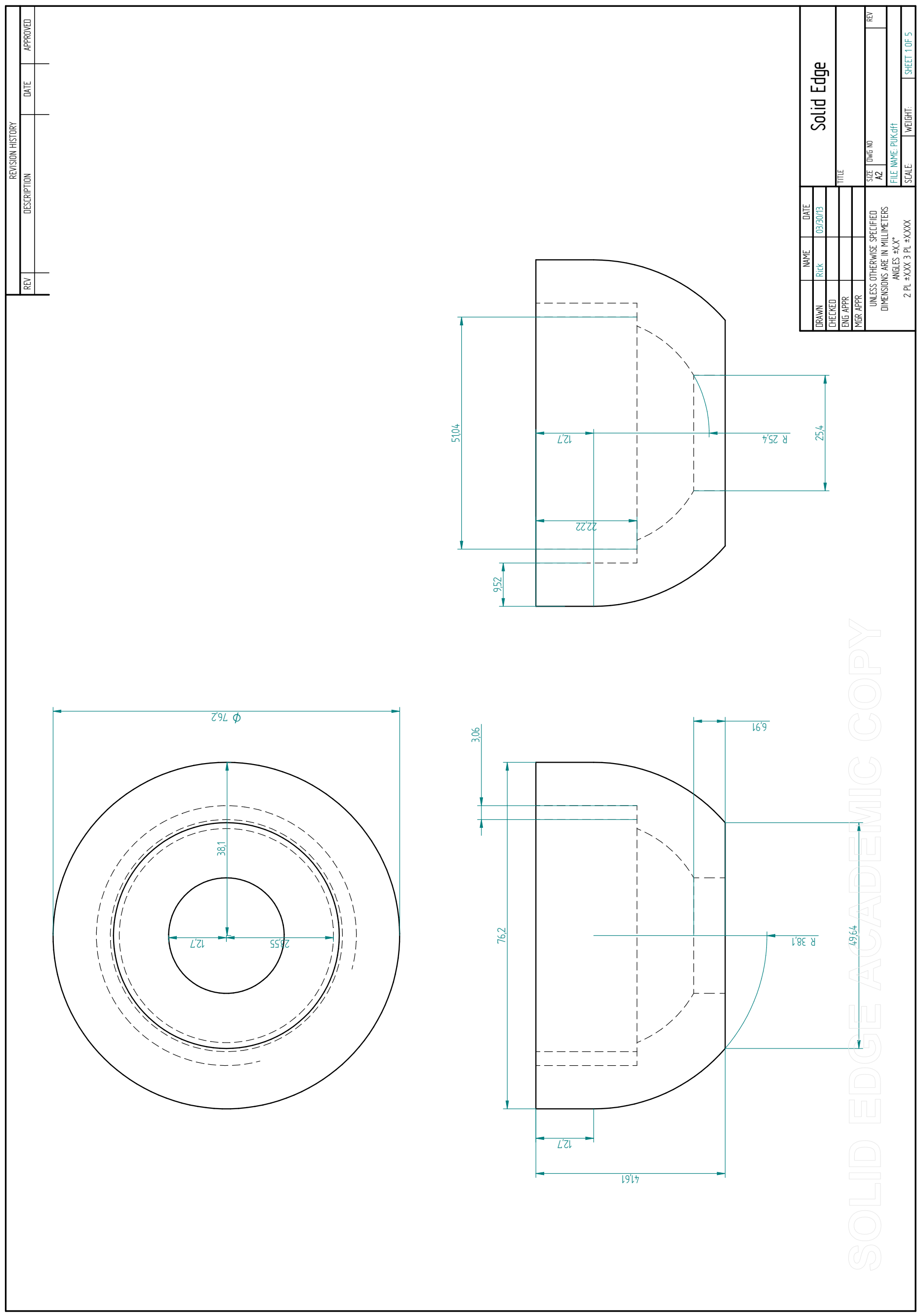
Appendix A

CAD Drawings

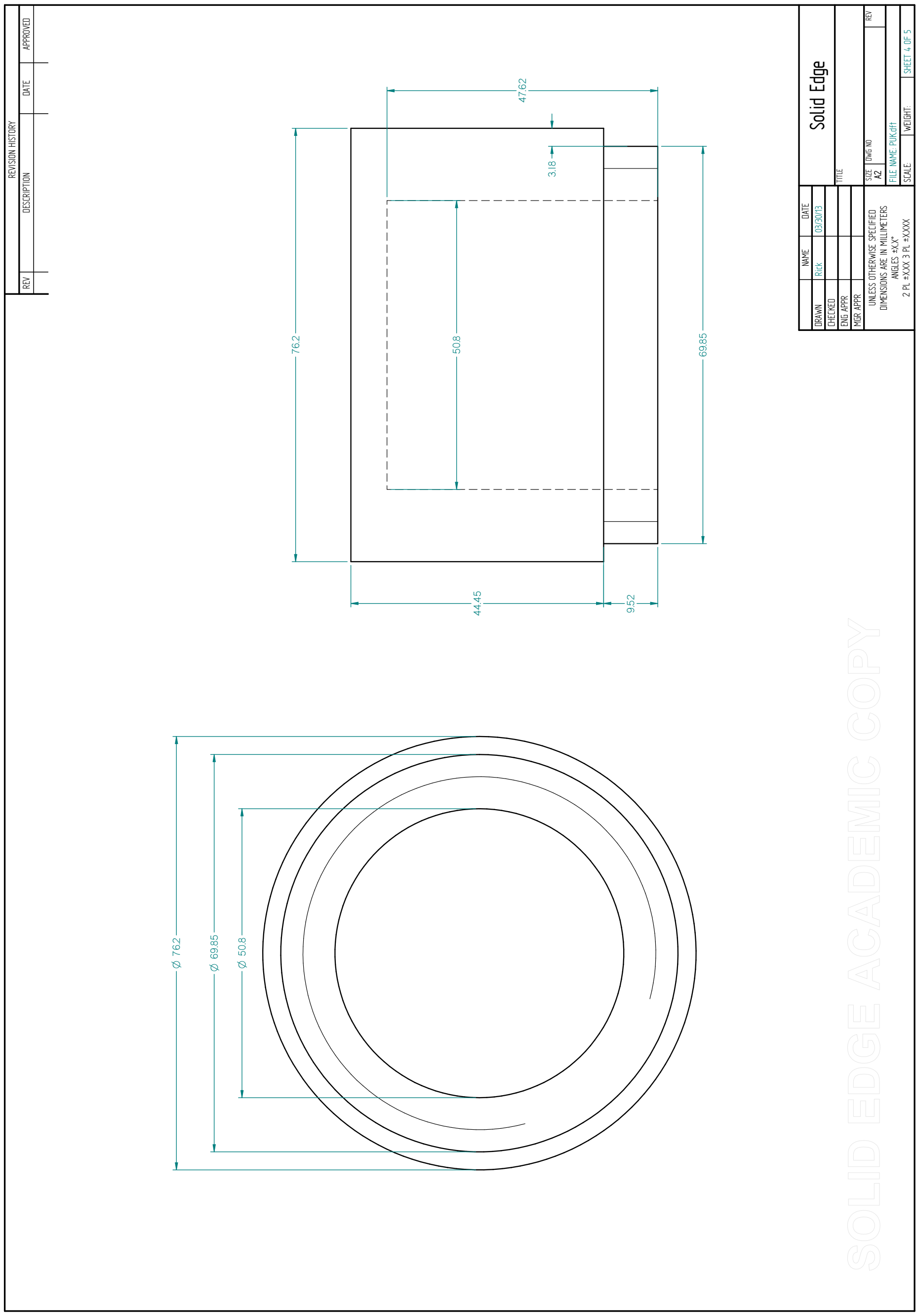
A.1 Technical Layout / Assembly



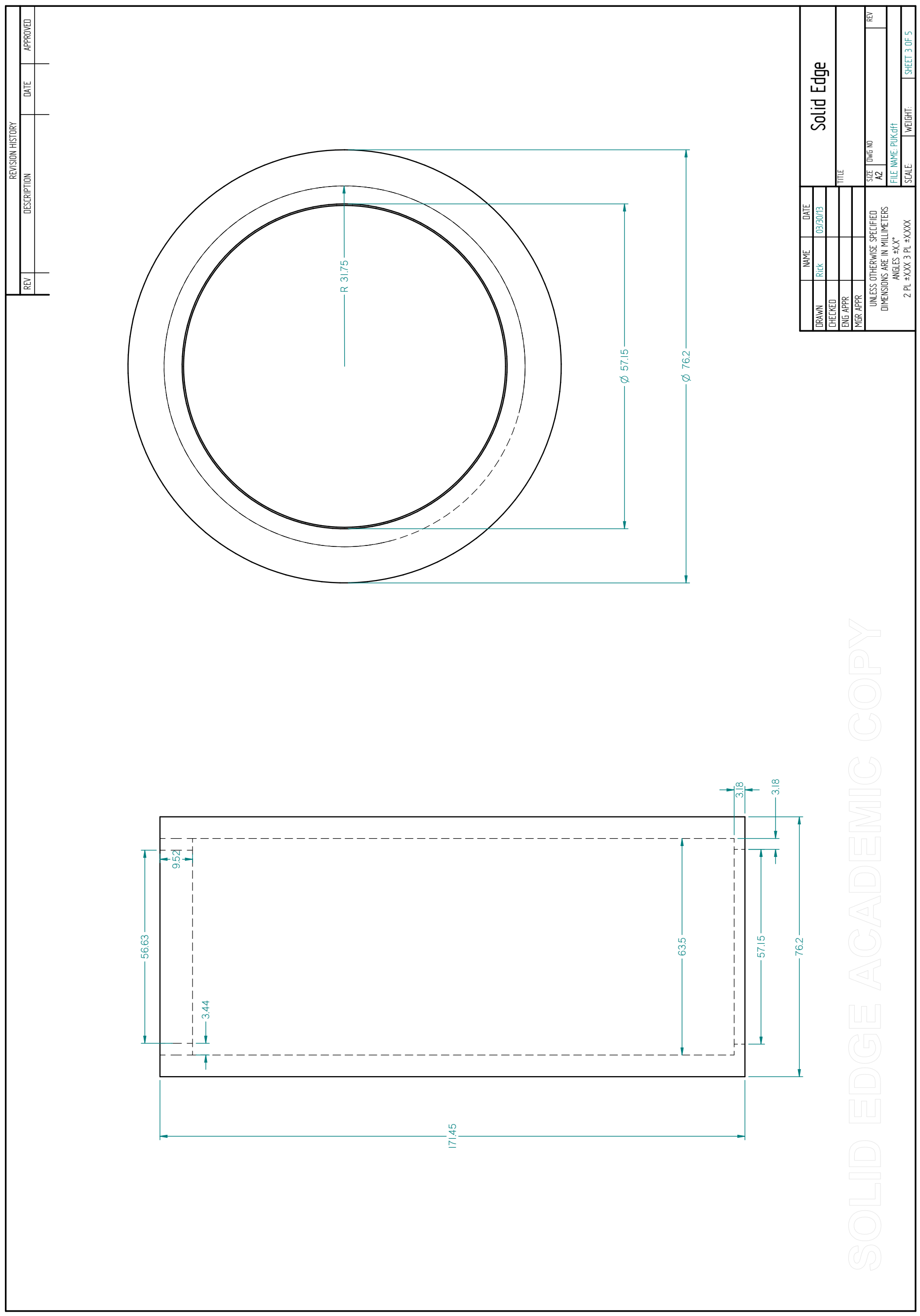
A.2 Lower Cap


REVISION HISTORY
REV
DESCRIPTION
DATE
APPROVED
Solid Edge
TITLE
DRAWN
CHECKED
ENG APPR
MGR APPR
NAME
DATE
UNLESS OTHERWISE SPECIFIED
DIMENSIONS ARE IN MILLIMETERS
SIZE
A2
DWG NO
REV
SCALE:
WEIGHT:
SHEET 1 OF 5
50,04
12,7
R 25,4
25,4
22,22
9,52
Φ 76,2
38,1
12,7
25,55
3,06
6,91
76,2
R 38,1
49,64
12,7
41,61
SOLID EDGE ACADEMIC COPY

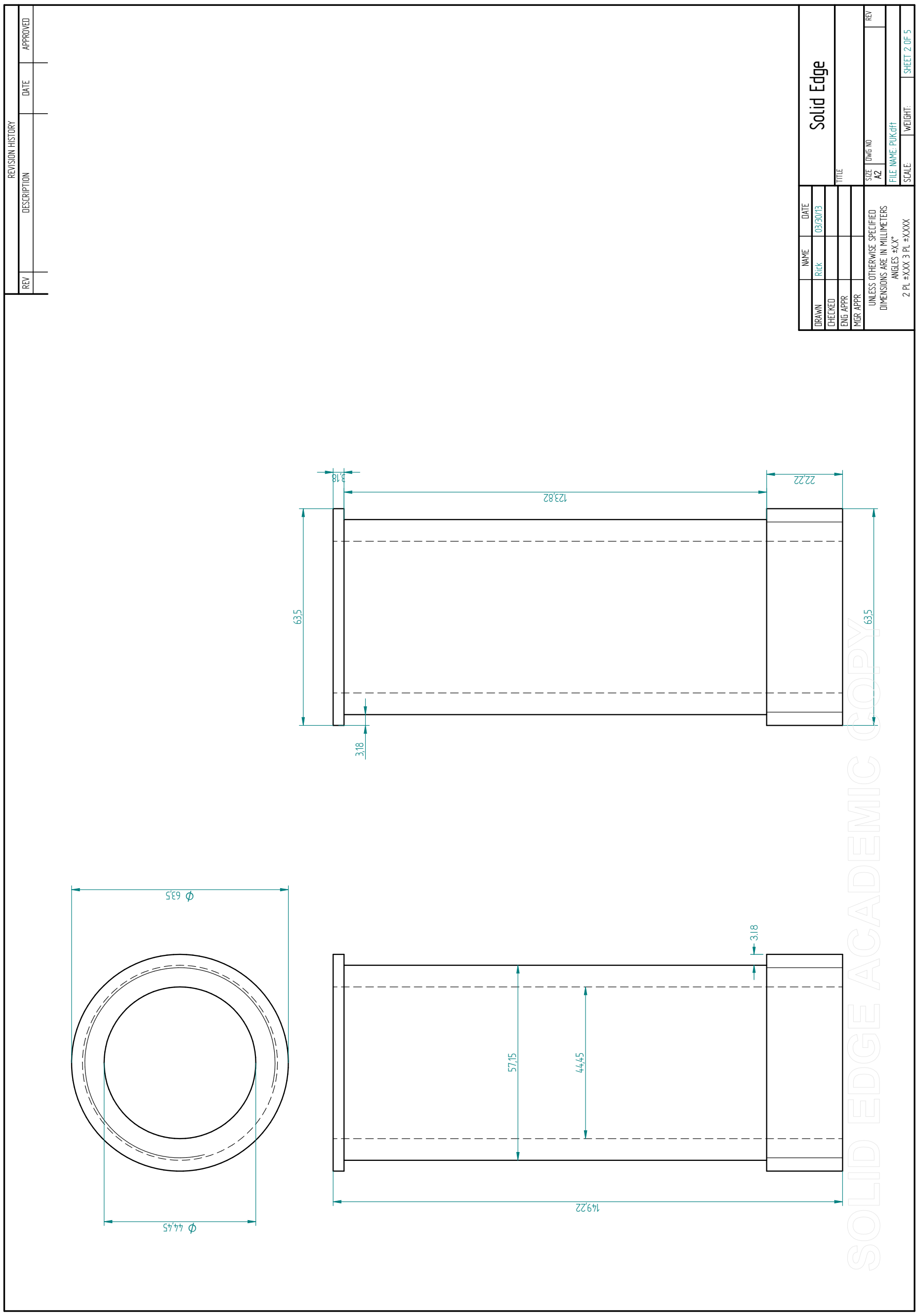
A.3 Upper Cap


47.62
3.18
76.2
50.8
69.85
44.45
9.52
Ø 76.2
Ø 69.85
Ø 50.8
REVISION HISTORY
REV
DESCRIPTION
DATE
APPROVED
Solid Edge
TITLE
NAME
DATE
DRAWN
Rick
CHECKED
ENG APPR
MGR APPR
UNLESS OTHERWISE SPECIFIED
DIMENSIONS ARE IN MILLIMETERS
SIZE
A2
DWG NO
REV
FILE NAME: PUK.dft
SCALE:
WEIGHT:
SHEET 4 OF 5
SOLID EDGE ACADEMIC COPY

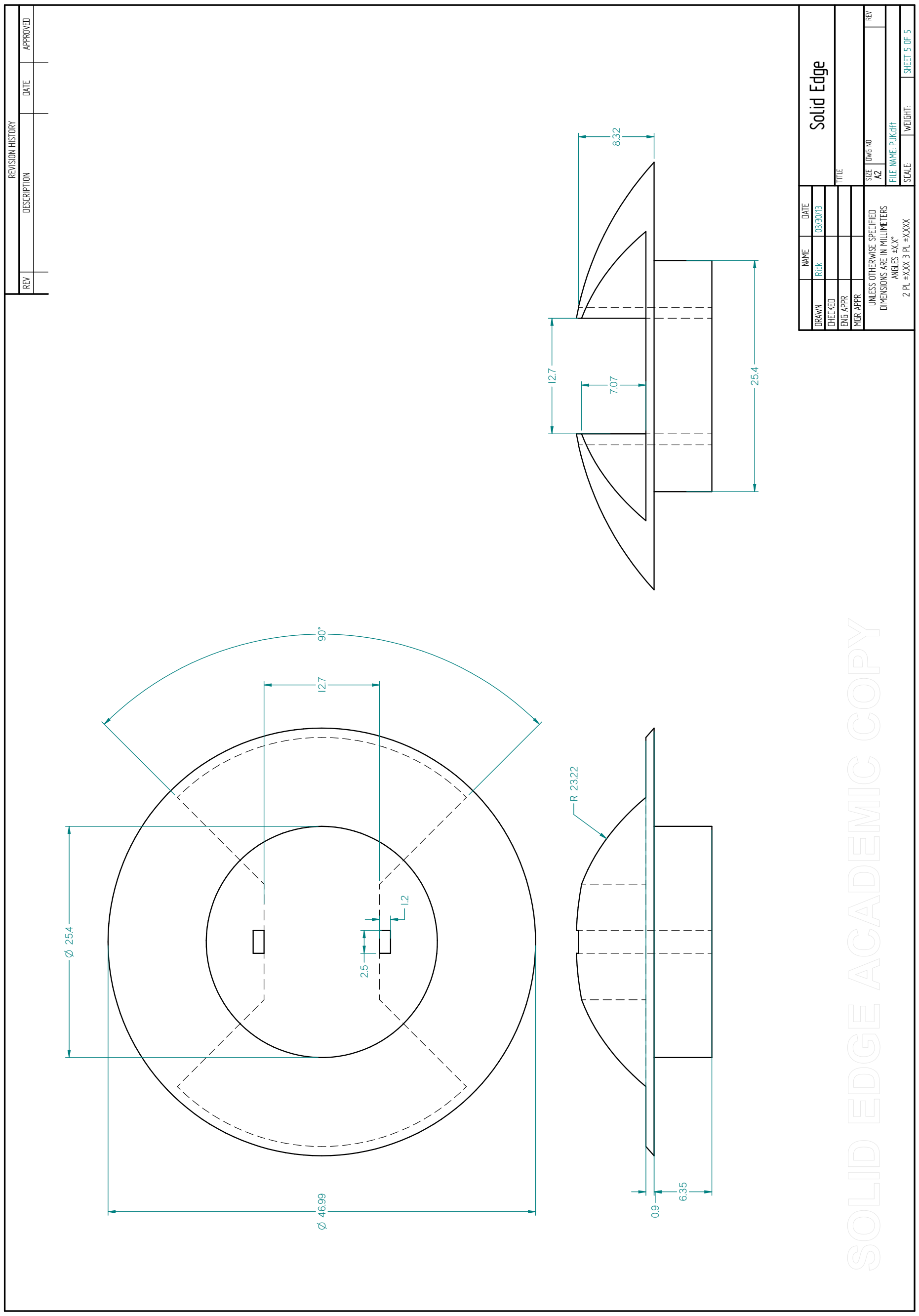
A.4 Upper Tube



A.5 Lower Tube



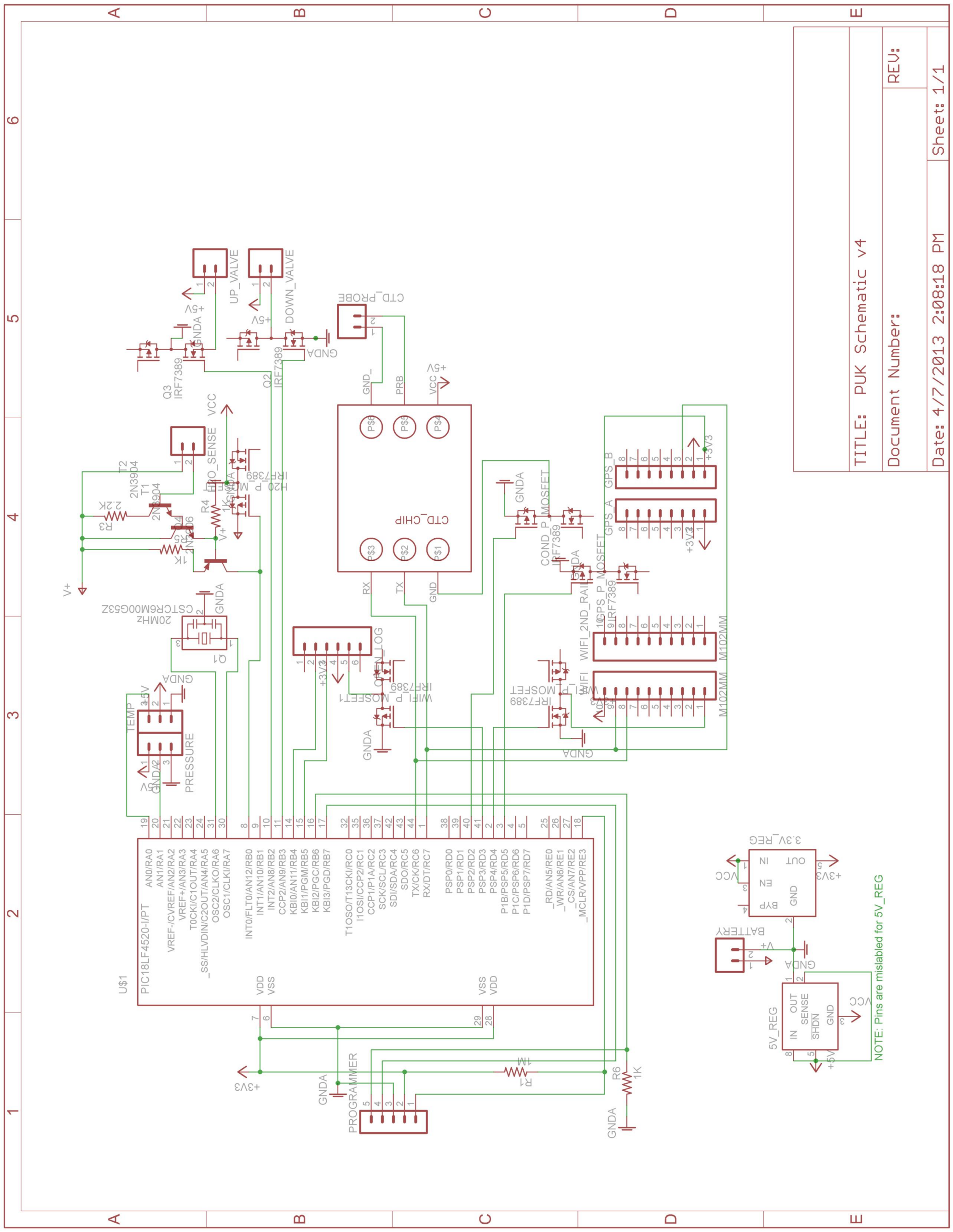
A.6 Probe Cap


REVISION HISTORY
REV
DESCRIPTION
DATE
APPROVED
Solid Edge
8.32
12.7
7.07
25.4
90°
12.7
R 23.22
Ø 25.4
1.2
2.5
Ø 46.99
0.9
6.35
SOLID EDGE ACADEMIC COPY

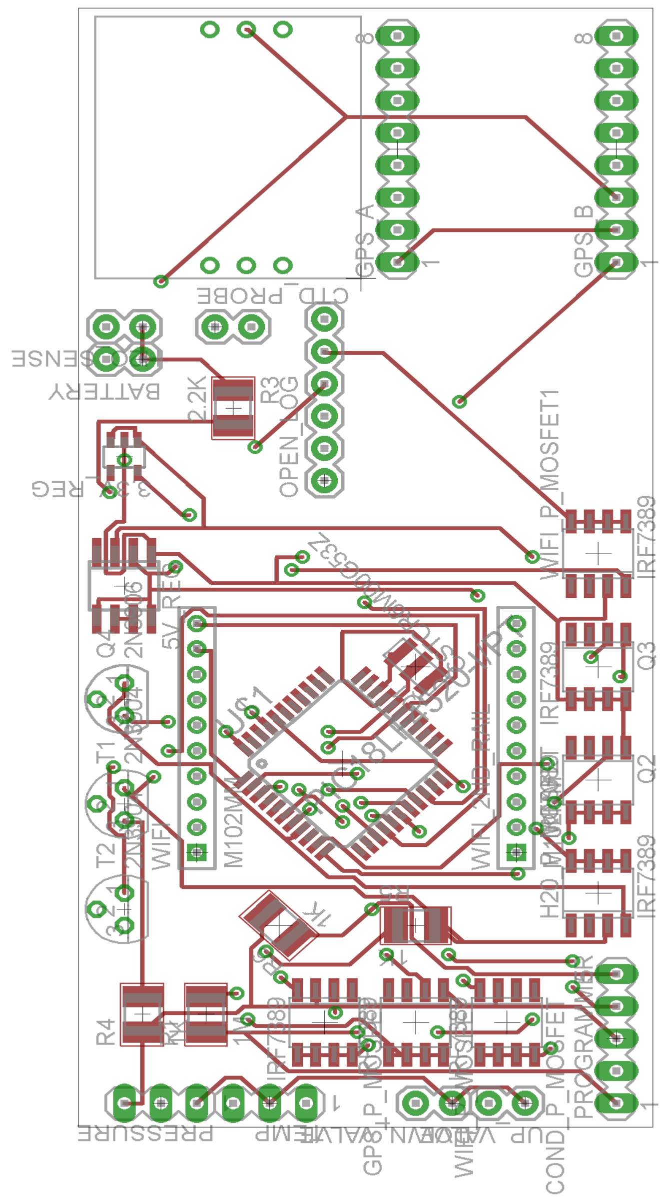
Appendix B

Circuit Board Schematics

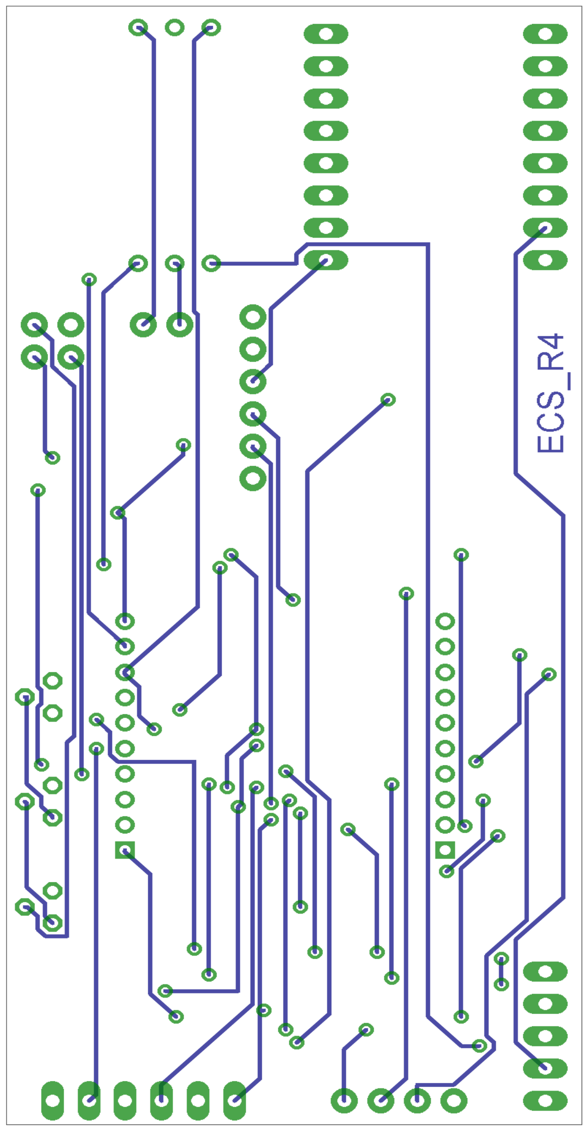
B.1 Schematic



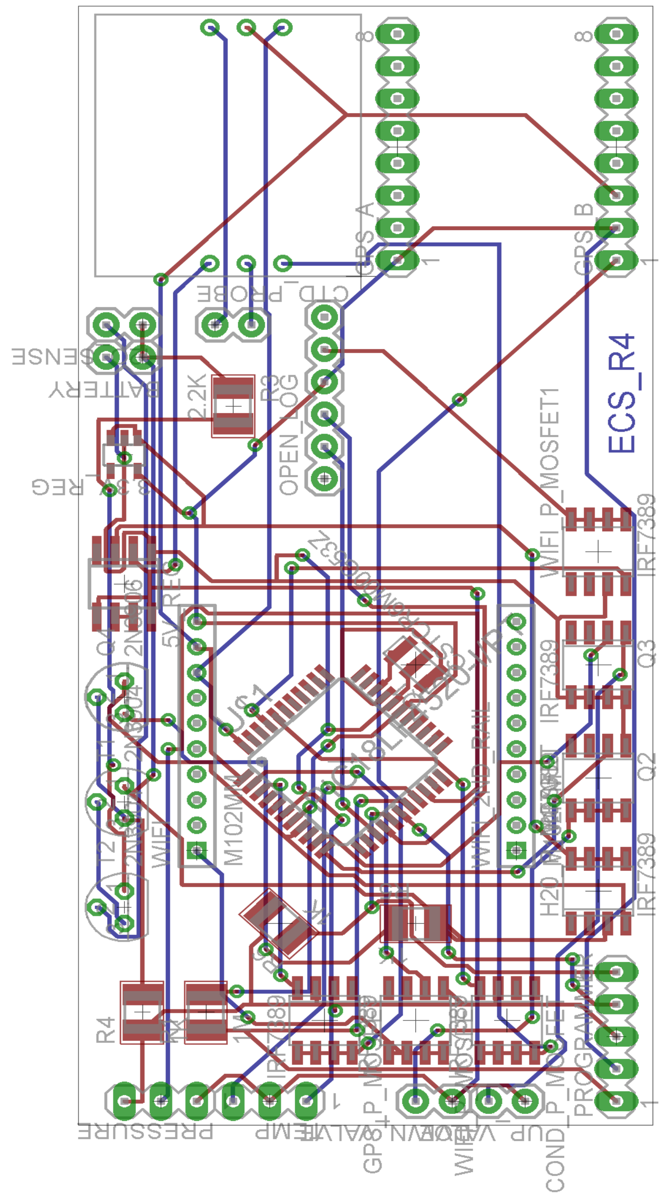
B.2 Board Top Layer



B.3 Board Bottom Layer



B.4 Complete Board Layout



B.5 Circuit Board Parts List

```
Partlist

Exported from PUK Schematic v4.sch at 4/30/2013 8:57:20 AM

EAGLE Version 6.2.0 Copyright (c) 1988-2012 CadSoft

Assembly variant:

Part           Value             Device            Package  Library
Sheet

3.3V_REG                         MIC5219XX         SOT23-5  v-reg-micrel      1
5V_REG                           LT1129CS8         SOIC8    linear-technology 1
BATTERY                          M02PTH            1X02     SparkFun          1
COND_P_MOSFET  IRF7389           IRF7389           SO08-IRF transistor-fet    1
CTD_PROBE                        M02PTH            1X02     SparkFun          1
DOWN_VALVE                       M02PTH            1X02     SparkFun          1
GPS_A                            MA08-1            MA08-1   con-lstb          1
GPS_B                            MA08-1            MA08-1   con-lstb          1
GPS_P_MOSFET   IRF7389           IRF7389           SO08-IRF transistor-fet    1
H2O_SENSE                        M02PTH            1X02     SparkFun          1
H20_P_MOSFET   IRF7389           IRF7389           SO08-IRF transistor-fet    1
OPEN_LOG                         M06SIP            1X06     SparkFun          1
PRESSURE                         MA03-1            MA03-1   con-lstb          1
PROGRAMMER                       MA05-1            MA05-1   con-lstb          1
Q1             CSTCR6M00G53Z     CSTCR6M00G53Z     CSTCR6M  murata-filter     1
Q2             IRF7389           IRF7389           SO08-IRF transistor-fet    1
Q3             IRF7389           IRF7389           SO08-IRF transistor-fet    1
Q4             2N3906            2N3906            TO92     transistor-pnp    1
R1             1M                R-US_R1210        R1210    rcl               1
R3             2.2K              R-US_R1210        R1210    rcl               1
R4             1K                R-US_R1210        R1210    rcl               1
R5             1K                R-US_R1210        R1210    rcl               1
R6             1K                R-US_R1210        R1210    rcl               1
T1             2N3904            2N3904            TO92     transistor        1
T2             2N3904            2N3904            TO92     transistor        1
TEMP                             MA03-1            MA03-1   con-lstb          1
U$1            PIC18LF4520-I/PT  PIC18LF4520-I/PT  TQFP44   pic18fxx20        1
U$2            CTD_CHIP          CTD_CHIP          CTD_CHIP CTD_CHIP          1
UP_VALVE                         M02PTH            1X02     SparkFun          1
WIFI           M102MM            M102MM            1X10-2MM SparkFun          1
WIFI_2ND_RAIL  M102MM            M102MM            1X10-2MM SparkFun          1
WIFI_P_MOSFET  IRF7389           IRF7389           SO08-IRF transistor-fet    1
WIFI_P_MOSFET1 IRF7389           IRF7389           SO08-IRF transistor-fet    1
```

Appendix C

Source Code

C.1 Chip Config Code

```
// PIC18F4520 Configuration Bit Settings

#include <p18cxxx.h>

// CONFIG1H
#pragma config OSC = HS   // Oscillator Selection bits (HS oscillator)

#pragma config FCMEN = OFF      // Fail-Safe Clock Monitor Enable bit (Fail-Safe Clock
Monitor disabled)

#pragma config IESO = OFF       // Internal/External Oscillator Switchover bit (Oscillator
Switchover mode enabled)


// CONFIG2L
#pragma config PWRT = OFF       // Power-up Timer Enable bit (PWRT disabled)

#pragma config BOREN = OFF      // Brown-out Reset Enable bits (Brown-out Reset disabled in
hardware and software)

#pragma config BORV = 3         // Brown Out Reset Voltage bits (Minimum setting)


// CONFIG2H
#pragma config WDT = OFF        // Watchdog Timer Enable bit (WDT disabled (control is
placed on the SWDTEN bit))

#pragma config WDTPS = 32768    // Watchdog Timer Postscale Select bits (1:32768)


// CONFIG3H
#pragma config CCP2MX = PORTC   // CCP2 MUX bit (CCP2 input/output is multiplexed with
RC1)

#pragma config PBADEN = ON      // PORTB A/D Enable bit (PORTB<4:0> pins are configured
as analog input channels on Reset)

#pragma config LPT1OSC = OFF    // Low-Power Timer1 Oscillator Enable bit (Timer1
configured for higher power operation)

#pragma config MCLRE = ON       // MCLR Pin Enable bit (MCLR pin enabled; RE3 input pin
disabled)


// CONFIG4L
```

```
#pragma config STVREN = OFF     // Stack Full/Underflow Reset Enable bit (Stack
full/underflow will not cause Reset)

#pragma config LVP = OFF        // Single-Supply ICSP Enable bit (Single-Supply ICSP enabled)

#pragma config XINST = OFF      // Extended Instruction Set Enable bit (Instruction set
extension and Indexed Addressing mode disabled (Legacy mode))


// CONFIG5L
#pragma config CP0 = OFF        // Code Protection bit (Block 0 (000800-001FFFh) not code-
protected)

#pragma config CP1 = OFF        // Code Protection bit (Block 1 (002000-003FFFh) not code-
protected)

#pragma config CP2 = OFF        // Code Protection bit (Block 2 (004000-005FFFh) not code-
protected)

#pragma config CP3 = OFF        // Code Protection bit (Block 3 (006000-007FFFh) not code-
protected)


// CONFIG5H
#pragma config CPB = OFF        // Boot Block Code Protection bit (Boot block (000000-
0007FFh) not code-protected)

#pragma config CPD = OFF        // Data EEPROM Code Protection bit (Data EEPROM not
code-protected)


// CONFIG6L
#pragma config WRT0 = OFF       // Write Protection bit (Block 0 (000800-001FFFh) not write-
protected)

#pragma config WRT1 = OFF       // Write Protection bit (Block 1 (002000-003FFFh) not write-
protected)

#pragma config WRT2 = OFF       // Write Protection bit (Block 2 (004000-005FFFh) not write-
protected)

#pragma config WRT3 = OFF       // Write Protection bit (Block 3 (006000-007FFFh) not write-
protected)


// CONFIG6H
```

#pragma config WRTC = OFF // Configuration Register Write Protection bit (Configuration registers (300000-3000FFh) not write-protected)

#pragma config WRTB = OFF // Boot Block Write Protection bit (Boot block (000000-0007FFh) not write-protected)

#pragma config WRTD = OFF // Data EEPROM Write Protection bit (Data EEPROM not write-protected)

// CONFIG7L
#pragma config EBTR0 = OFF // Table Read Protection bit (Block 0 (000800-001FFFh) not protected from table reads executed in other blocks)

#pragma config EBTR1 = OFF // Table Read Protection bit (Block 1 (002000-003FFFh) not protected from table reads executed in other blocks)

#pragma config EBTR2 = OFF // Table Read Protection bit (Block 2 (004000-005FFFh) not protected from table reads executed in other blocks)

#pragma config EBTR3 = OFF // Table Read Protection bit (Block 3 (006000-007FFFh) not protected from table reads executed in other blocks)

// CONFIG7H
#pragma config EBTRB = OFF // Boot Block Table Read Protection bit (Boot block (000000-0007FFh) not protected from table reads executed in other blocks)

C.2 Main Code

```
/*
 * File:   Puk_main.c
 * Author: Mathew Jordan and Rick Paridis
 *
 * Created on May 19, 2012, 4:02 PM
 */

//TO DO code in a command prompt
//---------------------------------------------

#include <p18f4520.h>
#include <stdio.h>
#include <stdlib.h>
#include <usart.h>
#include <delays.h>
#include <math.h>
#include <sw_uart.h>
#include "puk_globals.h"
#include "puk_read_ops.h"
#include "puk_tx_ops.h"
#include "puk_write_ops.h"
#include "puk_cmd.h"
#include "puk_salinity.h"


#pragma code isr=0x08
#pragma interrupt isr

//fuction prototypes
void parse_gps(void);
void surface(void);
void dive(void);
float get_depth(void);
void sleep(long int stime);
void isr(void)
{
   INTCONbits.GIE = 0;
   // interrupt handling code here
   if (TMR1_int)
   {
      TMR1_int = 0;
      TMR1_reset;
      get_flag++;
   }
   if (TMR0_int)
   {
```

```
        TMR0_int = 0;
        TMR0_reset;
        time++;
    }

    if (PIR1bits.RCIF)
    {
        recv_buff[n] = RCREG;
        if (recv_buff[n] == 0x0a)parse_flag = 1;cmd_flag=2;
        if (recv_buff[n]== '@')
        {
            cmd_flag=1;
            recv_buff[n]=0;
        }
        if (recv_buff[n]==13)
        {
            ctd_flag=1;
        }
        if (recv_buff[n++] == '$')n = 1;

    }

    INTCONbits.GIE = 1;
}
#pragma code

//-------------------------------------------------------------
// Begin Main Code
//-------------------------------------------------------------

void main(void)
{
    unsigned char u = 0;

    // set ports
    ADCON0 = 0b00000001;
    ADCON1 = 0b00001101;
    ADCON2 = 0b10011011;
    TRISA = 0b11111111;
    PORTA = 0;
    TRISB = 0b00000000;
    PORTB = 0;
    TRISD= 0;

    LOG_EN=1;
```

```
    //set interupts
    INTCON = 0b01000000; //timer 0
    PIE1 = 0b00100000; //timer 1

    OpenUSART(USART_TX_INT_OFF &
            USART_RX_INT_ON &
            USART_ASYNCH_MODE &
            USART_EIGHT_BIT &
            USART_CONT_RX &
            USART_BRGH_HIGH, 129); //9600 baud 25 High 4mhz
                        //129 high 20 mhz
                        //64 high 8 mhz
    OpenUART();

    T1CON = 0b10101001; //16 bit, 8 prescaler

    INTCONbits.GIE = 1;
    TMR1_on;

    Delay10KTCYx(100);

    POWER=1;

// Begin Main Loop
    while (1)
    {
#ifndef DEBUG
        if (data.depth<0)  //While at Surface
        {
            TMR1_off;
            tx_data();
            //TODO new file for logger dive number
        }
        else
        {
            TMR1_on;
        }
        if(data.depth>10.0)  // When Max Depth Reached
        {
            data.depth=data.depth/(float)data.count;
            surface();
        }
#else
        TMR1_on;
        LOG_EN=1;
#endif
```

```
// Call Data Read for Depth and Temp
data.temp=data.temp+get_temp();
data.depth=data.depth+get_depth();
data.count++;
   //Command Prompt Code Here
   //        if (cmd_flag)
   //        {
   //            GPS_EN=0;
   //            cmd_promt();
   //        }

if (get_flag>=RATE) //5 2hz for 20 mhz
{

   GPS_EN=1;
   LOG_EN=1;
   data.temp=data.temp/(float)data.count;
   data.depth=data.depth/(float)data.count;
   data.count=0;

   // Call Get Salinity for Salinity and Conductivity
   get_salinity(data.temp);
   RADIO_EN=1;

Delay10KTCYx(100);
// Delay10KTCYx(100);
   get_flag = 0;
   //CloseUSART();
   write_data();
   SENSOR_EN=0;

   tx_data();
   RADIO_EN=0;
   SENSOR_EN=1;
   LOG_EN=0;
   // UP=1;      //Uncomment to activate Dive Sequence again
   //sleep(1);   //Uncomment to activate Sleep Cycle
   // UP=0;      //Uncomment to activate Dive Sequence again
   data.temp=0.0;
   data.depth=0.0;


}
   // Get GPS Reading
if (parse_flag)
```

```
        {
            parse_gps();
            for (u = 0; u <= buff; u++)recv_buff[u] = '\0';
            parse_flag = 0;
        }

    }

}
//--------------------------------------------------------------
// Functions
//--------------------------------------------------------------
void parse_gps(void)
{
    unsigned char i = 0, pos = 0, rmc = 0, q = 0, comma = 0, count = 0;
    char raw[10][10] = {"\0"}, stuff[10] = {"\0"};

    for (pos = 0; pos < buff; pos++)
    {
        if (recv_buff[pos] == 'R')
        {
            rmc = 1;
        }
    }
    if (rmc)
    {
        for (pos = 0; pos < buff; pos++)
        {
            if (recv_buff[pos] == 0x2c)
            {
                pos++;
                while (recv_buff[q + pos] != 0x2c)
                {
                    raw[comma][q] = recv_buff[q + pos];
                    q++;
                }
                comma++;
                q = 0;
            }
        }
        for (count = 0; count < 12; count++)
        {
           stuff[count] = '\0';
        }
        for (count = 0; count < 9; count++)
        {
```

```
            stuff[count] = raw[0][count];
        }

        data.time = atof(stuff);


        for (count = 0; count <= 10; count++)stuff[count] = raw[1][count];
        data.valid = stuff[0];
        if (data.valid == 'V')return;

        for (count = 0; count <= 10; count++)stuff[count] = raw[2][count];
        data.lat = atof(stuff);

        for (count = 0; count <= 10; count++)stuff[count] = raw[3][count];
        data.ns = stuff[0];

        for (count = 0; count <= 10; count++)stuff[count] = raw[4][count];
        data.longi = atof(stuff);

        for (count = 0; count <= 10; count++)stuff[count] = raw[5][count];
        data.ew = stuff[0];

        for (count = 0; count <= 10; count++)stuff[count] = raw[8][count];
        data.date = atol(stuff);

    }

}

void dive(void)
{
    GPS_EN = 0;
    RADIO_EN = 0;
    TMR0_reset;
    TMR0_on;
    DOWN = 1;
    while (time < 2);
    TMR0_off;
    time = 0;
    DOWN = 0;

}

void surface(void)
{
    float p1 = 0, p2 = 0;
```

```
    UP = 1;
    p1 = data.depth;
    p2 = p1;
    while (p2 + .1 > p1)
    {
        p2 = get_depth();
        Delay1KTCYx(100);
        UP = !UP;
    }
    UP = 0;
}

float get_depth(void)
{
    float v = 0;
    //set channel AN1
    ADCON0 = 0b00000111;
    ADCON0bits.GO = 1;
    while (ADCON0bits.GO);
    v = ADRES * 5.0 / 1024.0;
    v = v - 0.2;
    press = v / 0.0064;
    return press * 1000.0 / 10074.87;
}

void sleep( long int stime)
{
    long int T=0;
    //TODO sleep for time
    GPS_EN=0;
    RADIO_EN=0;
    SENSOR_EN=0;
    for(T=0;T<=stime;T++)
    {
       Sleep();
    }
    GPS_EN=1;
    RADIO_EN=1;
    SENSOR_EN=0;
}
```

C.3 Globals Code

```
/*
This Code initializes the Global Variables and Delays, and inserts them
into a structure to easily call them in the Main Code
*/

#include <delays.h>

unsigned char n = 0, parse_flag = 0, get_flag = 0;

char recv_buff[100], cmd_flag = 0, ctd_flag=0;

int time = 0;

float press = 0, ctd_v = 0, ctd_current = 0;

struct status {    // This structure holds the status variables for the Main Code

    unsigned char surfacing;

    unsigned char diving;

    unsigned char transmitting;

} status;

struct data {    // This structure holds the data variables for the Main Code

    float time;

    char valid;

    float lat;

    char ns;

    float longi;

    char ew;

    long int date;

    float depth;

    float depth_2;

    float temp;

    char atlas[15];

    int count;

} data;
```

C.4 Globals Code

```
void DelayTXBitUART (void){
   Delay10TCYx(50);
   Nop();
   Nop();
   Nop();
   Nop();
   Nop();
   Nop();
   Nop();
   Nop();
   Nop();

}
void DelayRXHalfBitUART (void){
   Delay10TCYx(25);
   Nop();
   Nop();
}
void DelayRXBitUART (void){
   Delay10TCYx(50);
   Nop();
   Nop();
   Nop();
   Nop();
   Nop();
   Nop();
```

C.5 Globals Code

```
    Nop();
}
void CloseUART(void)
{
    TRISBbits.RB4=1;
    TRISBbits.RB5=1;
}
```

C.6 Data Read Code

```
/*
This code reads the data through functions for the Main Code
*/

#include <p18f4520.h>
#include <usart.h>
#include <delays.h>
#include "puk_globals.h"
#include "puk_read_ops.h"

void read_data(void) {
    data.time = read_float();
    data.lat = read_float();
    data.longi = read_float();
    data.depth = read_float();
    data.temp = read_float();
    //data.salinity = read_float();  //This will be done in Get Salinity Code instead
}

float read_float(void) {
}
long read_long(void) {
}
```

C.7 Get Salinity Code

```
/*
This Code Reads the Temp of the water and then sends
that value to the Atlas Chip to receive a temperature
compensated salinity reading.  Upon receiving the reading
the pic chip sends the values to the OpenLog for data logging
*/


#include <p18f4520.h>

#include <delays.h>

#include <math.h>

#include <sw_uart.h>

#include "puk_globals.h"

#include "puk_salinity.h"

#include <stdlib.h>

#include <stdio.h>

#include <usart.h>

// Read in temperature to receive temp compensated salinity
float get_temp(void)

{

   float v = 0;

   //set channel AN0

   ADCON0 = 0b00000111;

   ADCON0bits.GO = 1;

   while (ADCON0bits.GO);

   v = ADRES * 3.12 / 1024.0;

   return 21.25;

   //return (v/0.010);

}
```

```
//  Read temp compensated Salinity
void get_salinity(float temp)

{

   char b[20]={'\0'};

   char i=0;

   int upper = 0;

   unsigned int lower = 0;

   for(i=0;i<16;i++)

   {

            data.atlas[i]=0;

         }


     Delay10KTCYx(100);

      Delay10KTCYx(100);

      Delay10KTCYx(100);

      Delay10KTCYx(100);

      Delay10KTCYx(100);

      Delay10KTCYx(100);

      Delay10KTCYx(100);

//Read values into the pic chip
      OpenUSART(USART_TX_INT_OFF &

              USART_RX_INT_ON &

              USART_ASYNCH_MODE &

              USART_EIGHT_BIT &

              USART_CONT_RX &

              USART_BRGH_HIGH, 32); //38400

      upper = (int) temp;

      lower = (temp - (float) upper)*100.0;
```

```
sprintf(b, "%d.%u\r",upper, lower);
i=0;
Delay10KTCYx(100);
Delay10KTCYx(100);
Delay10KTCYx(100);
INTCONbits.GIE = 1;
while(i<7)
{
   if(b[i]==0)
   {
      break;
   }
   while(BusyUSART());
   putcUSART(b[i]);
   i++;
}
n=0;
Delay10KTCYx(100);
Delay10KTCYx(100);
Delay10KTCYx(100);
Delay10KTCYx(100);
if(ctd_flag){
   for(i=n;i>-1;i--)
   {
      if(recv_buff[i]<200)
      {
      data.atlas[i]=recv_buff[i];
```

```
            }
        }
        ctd_flag=0;
    }
//Save reading into OpenLog
    OpenUSART(USART_TX_INT_OFF &
            USART_RX_INT_ON &
            USART_ASYNCH_MODE &
            USART_EIGHT_BIT &
            USART_CONT_RX &
            USART_BRGH_HIGH, 128); //38400
    Delay10KTCYx(100);
    Delay10KTCYx(100);
}
```

C.8 Data Save Code

```
/*
This code transfers the data received into the pic chip and
transmits it out to the OpenLog chip to be logged into a file
*/


#include <p18f4520.h>

#include <sw_uart.h>

#include <usart.h>

#include <delays.h>

#include<stdio.h>

#include<stdlib.h>

#include "puk_globals.h"

#include "puk_write_ops.h"

#include "puk_read_ops.h"

//#include "puk_tx_ops.h"


void write_data(void) {

   //valid,time,lat,long,depth,temp,conduct

   OpenUART();

   putcUART('$');

   putcUART(data.valid);

   putcUART(' ');

   write_float(data.time);

   Delay1KTCYx(100);

   write_float(data.lat);

   Delay1KTCYx(100);

   write_float(data.longi);

   Delay1KTCYx(100);
```

```
    write_float(data.depth);
    Delay1KTCYx(100);
    write_float(data.temp);
    Delay1KTCYx(100);
    putcUART(',');
    Delay1KTCYx(100);
    putsUART(data.atlas);
    Delay1KTCYx(100);
    putcUART(0x0A);
}

void write_float(float fdata) {
    char b[100] = {"\0"},i=0;
    long int upper = 0;
    unsigned long int lower = 0;
    OpenUART();
    upper = (long int) fdata;
    lower = (fdata - (float) upper)*1000.0;
    if (lower >= 100)
    {
        sprintf(b, ",%ld.%lu", upper, lower);
    }
    else if (lower >= 10 && lower < 100)
    {
        sprintf(b, ",%ld.0%lu", upper, lower);
    }
    else if (lower < 10)
```

```
    {
        sprintf(b, ",%ld.00%lu", upper, lower);
    }
    while(b[i]!=0)
    {
        WriteUART(b[i]);
        i++;
    }
}

void write_long(long ldata) {
    char b[100] = {"\0"};
    sprintf(b,",%lu",ldata);
    putsUART(b);
}
```

C.9 Data Transmit Code

```
/*
Similarly to the Data Save Code, this code transfers the data received
into the pic chip and transmits it out to the WIFI chip to be read through
into a computer through the dongle and logged into a file
*/

#include <p18f4520.h>
#include <usart.h>
#include <sw_uart.h>
#include <delays.h>
#include <i2c.h>
#include <stdlib.h>
#include <stdio.h>
#include "puk_globals.h"
#include "puk_read_ops.h"
#include "puk_tx_ops.h"


void tx_data(void)
{
   CloseUART();
    OpenUSART(USART_TX_INT_OFF &
              USART_RX_INT_ON &
              USART_ASYNCH_MODE &
              USART_EIGHT_BIT &
              USART_CONT_RX &
              USART_BRGH_HIGH, 128); //9600
   Delay10KTCYx(100);
   read_data();
```

```
    putcUSART('#');
    tx_float(data.time);
    tx_float(data.lat);
    tx_float(data.longi);
    tx_float(data.depth);
    tx_float(data.temp);
    putsUSART(data.atlas);
    putcUSART(0x0a);
}

void tx_float(float fldata)
{
    char b[100] = {"\0"};
    long int upper = 0;
    unsigned long int lower = 0;
    upper = (long int) fldata;
    lower = (fldata - (float) upper)*1000.0;
    if (lower >= 100)
    {
        sprintf(b, ",%ld.%lu", upper, lower);
    }
    else if (lower >= 10 && lower < 100)
    {
        sprintf(b, ",%ld.0%lu", upper, lower);
    }
    else if (lower < 10)
    {
```

```
        sprintf(b, ",%ld.00%lu", upper, lower);
    }
    putsUSART(b);
}
void tx_long(long ldata)
{
    char b[100] = {"\0"};
    sprintf(b,",%lu",ldata);
    putsUSART(b);
}
```