

ECE 480 DESIGN TEAM 11
Electronic Parachute Deployment System
Final Report

April 23rd, 2014

Sponsor: Pete Semig (Texas Instruments)



Facilitator: Joydeep Mitra

Design Team Members:

Michikazu Aono
Max Cooper
Zane Crawford
Qiaosen Huang
Pedro Rodriguez
Jessica Shelby

EXECUTIVE SUMMARY

The electronic parachute deployment system replaces the traditional parachute deployment method on a large model rocket that employs the use of a multistage parachute. The system will allow for improved consistency and reliability over the traditional, timed fuse system. The system will be controlled by the Texas Instruments MSP430G2553 microcontroller in conjunction with an analog altimeter sensor, amplification of the sensor output, power regulator, and parachute deployment system. The team pursued low-cost innovative solutions in developing the electronic parachute deployment system.

Through several tests the team was able to ensure proper functionality of each of the components required for the project. Using light emitting diodes on the microcontroller, the team simulated tests to ensure the pressure was properly detected from the analog-to-digital converter. Afterwards, the team checked for change in the instrumentation amplifier, the microcontroller detecting a peak to deploy the drogue chute, and the distance at half the peak height from ground level for the main chute. The final step in the design was to fabricate a printed circuit board. The team then tested the final revision using a pressure chamber prior to launch.

For a complete test of the system, the team launched the electronic parachute deployment system in both a large-scale rocket and mid-size rocket. The rocket launches both failed to reach their expected average altitude. However, even in the failure of the launches, the designed system functioned as specified. Our system grants model rocket hobbyists with a cheap alternative to launch and safely recover a rocket.

Acknowledgement

The authors would like to acknowledge and extend their gratitude to the following individuals and companies who have made the completion of this project possible:

Peter Semig: Without Mr. Semig's assistance throughout the project the team would not have had the proper guidance required to complete the project. He met with the team on a weekly basis and provided invaluable input to improving both the design and testing procedure.

Joydeep Mitra: The team's facilitator, Professor Mitra, contributed to the success of the team through ensuring the project was on track. Professor Mitra also provided feedback on assignments throughout the semester to improve the team's technical communications.

Advanced Circuits: The team would not have been able to stay within budget without Advanced Circuits' willingness to provide student discounts on printed circuit boards. The team went through three revisions of the PCB and Advanced Circuits saved the team \$193.76.

Texas Instruments: Texas Instrument's also greatly contributed to the team's success through providing free samples on several components including microcontrollers, amplifiers, and voltage regulators.

Greg Mulder and Brian Wright: The Electrical and Computer Engineering service center helped the team complete the project through providing the necessary tools for testing proper functionality. They also made a printed circuit board for our team to test the functionality of different amplifiers throughout the semester.

Mike Mock and Tim Claycomb: Mr. Mock and Claycomb, Texas Instrument employees, helped the team by participating in team meetings with Mr. Semig and provided feedback to assist the team in improving our project.

Roxanne Peacock: Mrs. Peacock assisted the team through purchasing the materials needed for the project and emailing the team in a timely manner when the components arrived.

Chapter 1: Intro/Background.....	1
1.1 Objectives.....	1
1.2 Size.....	2
1.3 Housing.....	2
1.4 Accuracy.....	2
1.5 Competition.....	3
Chapter 2: Exploring the Solution Space.....	4
2.1 Design Specifications.....	4
2.2 FAST Diagram.....	4
2.3 Conceptual Designs.....	5
2.4 House of Quality.....	7
2.5 Estimated Prototype Cost.....	7
2.6 Gantt Chart.....	8
Chapter 3: Technical Description.....	9
3.1 Sensors.....	9
3.1.1 Digital Altimeter.....	9
3.1.2 Analog Altimeter.....	10
3.2 Analog Amplifier.....	12
3.2.1 Operational Amplifier.....	12
3.2.2 Instrumentation Amplifier.....	13
3.3 Microcontroller.....	17
3.3.1 Hardware.....	17
3.3.2 Software.....	18
3.4 Parachute Deployment Fuse Igniter.....	19
3.5 Power System.....	21
3.6 Printed Circuit Board.....	24
Chapter 4: Test Data.....	28
4.1 Straw Test.....	28
4.2 Elevator Test.....	29
4.3 Vacuum Chamber Test.....	30
4.4 Rocket Launch Test.....	32
Chapter 5: Final Cost, Schedule, Summary, and Conclusions.....	34
5.1 Cost.....	34
5.2 Schedule.....	35
5.3 Future Work.....	35
5.4 Conclusions of Design.....	36
Appendix 1: Technical Roles, Responsibilities, and Work Accomplished.....	37
Appendix 2: References.....	42
Appendix 3: Code, Schematics, and Additional Pictures.....	43

Chapter 1 - Introduction and Background

Model rockets have been and will continue to be a popular hobby among many of all ages. As rockets become larger and more technically advanced, their recovery becomes more crucial. Many model rocket recovery systems exist today, making it a competitive market. The scope of this project is to design, simulate, fabricate, test and demonstrate a multistage Electronic Parachute Deployment (EPD) system using Texas Instruments (TI) components. The project is designed to expose the team to analog circuit design and simulation, printed circuit board (PCB) design and fabrication, microcontroller programming, and hardware testing.

1.1 Objectives

The project essentially has two main phases that each involve different objectives. The first phase involves designing and testing a BoosterPack for TI's MSP430 LaunchPad. The LaunchPad is a general purpose evaluation kit that can contain a MSP430 microcontroller (MCU) that has a 14-pin or 20-pin configuration. It has USB capability that allows for easy programming and a 20-pin header that connects to the BoosterPack. The BoosterPack contains a sensor that measures the altitude. Before designing the PCB that acts as the BoosterPack, the team will research sensors along with other analog circuit components needed to complete the design. After design ideas are chosen, the team will continuously run simulations as updates and improvements are made until the final design is selected. The PCB is designed to include all the analog connections along with the 20-pin connectors that are used to connect to the LaunchPad. The MCU is used to convert the sensor's output and interpret the data using code developed by the team. The second phase of the project is to integrate the overall design into a model rocket. The team will conduct research about model rockets to learn how much space and weight the rocket's payload bay can handle. Also, the team will build and launch a model rocket to test their system in.

The general purpose of this system is to deploy parachutes at intended times to help with the aim and stability of the rocket's landing. The EPD system's objectives are to detect the rocket's maximum altitude and launch multiple parachutes. The design of the EPD system consists of multiple sections. An altimeter is used to measure the changing altitude of the rocket's trajectory. Essentially, the altimeter's output feeds into the MSP430 MCU which contains an analog-to-digital converter (ADC). The microcontroller detects when the rocket starts to descent and sends a signal to the ignition section. At this time the ignition system lights one of two fuses that deploys the first parachute. The MCU waits a certain amount of time and then signals the ignition system again, this time lighting the second fuse which launches the main parachute. The ignition of the second fuse concludes the process of the EPD system.

The main extent of the project is to build and test an EPD system prototype. However, to further validate its functionality, the electronics will be used in a model rocket. Standards are a part of ensuring the safe and proper use of the EPD system and must be considered because the team planned on launching the final system into the sky. There are currently no standards regulating the use or operation of EPD systems in model rockets. There are standards for model rockets, however, which influence design considerations for EPD systems to make them useful for model rocket users.

1.2 Size

The EPD system designed for this project needed to fit inside the rocket and not affect the trajectory of the rockets used to separate the stages of the model rocket. Two model rockets were used throughout the duration of the project. Both rockets are from Apogee Components and are defined, from Apogee Components, as either a level 2 or a level 3 model rocket. The first rocket used is the Rising Star and is a level 2, meaning it is less difficult to build than a level 3. The Rising Star has a clear 2.2in diameter payload bay that was used to help the team design the system. This rocket was ordered as a kit and built by the team so the team could better understand how rockets work. The Rising Star does not require an electronic recovery system so it was launched by itself for a practice launch and later launched again with the electronics in its bay. The second rocket, the level 3 Madcow Torrent, was borrowed from a previous team and repaired for a second use. The payload bay for this rocket is 4in in diameter. This rocket requires an electronic recovery system and was supposed to be used as the final test for this project. Other model rockets with payloads can use the EPD as a recovery system, provided it can fit in the diameter of the rocket's payload bay. Therefore, a company standard would look for small devices that can fit in as many payload bays as possible.

1.3 Housing

The system will be exposed to a wide range of age groups of varying skill sets and require the handling of highly combustible and explosive devices. The device is to be enclosed in a plastic housing to protect the users, especially younger children, from potential electrical shocks and lacerations from sharp edges found on the soldered components. In addition to protecting the user, the housing will provide protection to internal components from external environmental factors, such as moisture (specifically the altimeter is highly sensitive to moisture and is recommended not to get it wet), and physical shock experienced on impact. The housing used for this project included a mounting point for the battery and house the MSP430 LaunchPad and BoosterPack in a tight secure package. The housing ensures the system components remain connected during launch and the flight of the rocket. Heavy vibrations during the launch phase could have potentially damaged the soldering point and loosen up the connections, so a properly designed case significantly reduced such defects from occurring. In addition to the main housing cell, the plastic case was required to provide areas to contain black powder (or black powder substitutes such as pyrodex) used in the deployment of the parachutes. These compartments were isolated from the main system to prevent un-tinted discharge via accidental short circuiting of the main system, and protect the charges from moisture which could render them useless. With these safety considerations put in place on the physical system, a great majority of disasters were eliminated or significantly mitigated.

1.4 Accuracy

The EPD system needed to have few faults resulting in a high accuracy. A fault with the EPD system can cause catastrophic damage to the model rocket and the user, depending on the charge system that is used in conjunction with the system. If the EPD triggers early, the

charge can expand and launch part of the rocket, potentially knock down the stand, and damage users if it triggers before the user can evacuate the area around the rocket. Also, if it triggers in the air on the ascent, it can damage the rocket. To prevent these damages, the EPD system needed to have very good resolution with a large range to detect launch height. By making sure a large change in voltage was needed to trigger the EPD system, there was little chance for error on the ground and in the air.

1.5 Competition

EPD systems currently on the market are fairly diverse. Some of the more expensive EPD systems include additional features such as flight recording or GPS tracking. In this regard the EPD system is fairly basic; however there are still multiple models that offer similar functionality. The main criterion that our design hopes to achieve compared to the competition is cost. The lowest price found on competing EPD systems was \$80. With our design we will try and minimize cost while giving the same level of performance of basic dual-deployment EPD systems on the market.

Chapter 2 - Exploring the Solution Space and Selecting a Specific Approach

2.1 Design Specifications

Prior to the team determining key components and features to incorporate in an electronic parachute deployment system, it was critical to determine the customer expectations and requirements for a successful project. Through several teleconference meetings with the Texas Instruments sponsor representative, Peter Semig, the team developed the following list indicating what was expected of a successful project and product.

The prototype device is...

- to be built around a TI MSP430 microcontroller
- to be a BoosterPack which mounts to the TI MSP430 LaunchPad
- equipped with test nodes for debugging
- to be powered by a battery
- to fit inside a model rocket
- to successfully deploy parachutes after the rocket reaches apogee
- reusable after a successful deployment

2.2 FAST Diagram

Once time had been spent to digest the project description and customer requirements the team set to decompose the system into subsystems defined by the role it plays in providing the essential feature (deploying a parachute). To assist in this noble quest the team employed the Function Analysis System Technique (FAST) diagram, Figure 2.1, to help visualize the functions. On the left side of the FAST diagram is the primary function the system is to perform, in this case deploy a parachute. To the right the primary function stems out the various supportive functions that must take place for the primary function to occur.

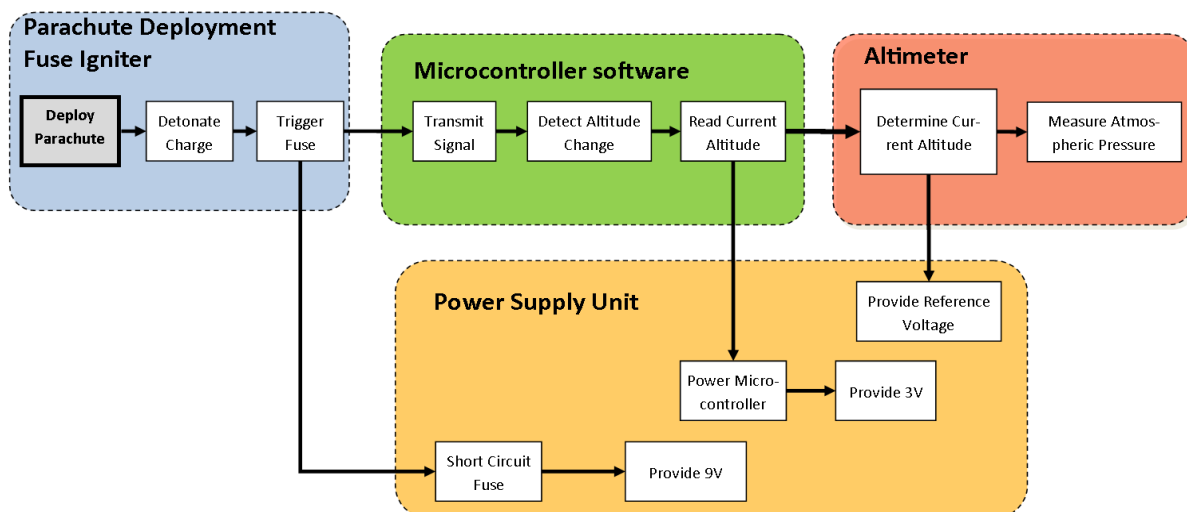


Figure 2.1 FAST Diagram

2.3 Conceptual Designs

After the team had identified the essential functions, both primary and secondary, the device was divided into subsystems defined by groups of related functions identified by the FAST diagram. These functions are listed below

I. Parachute Deployment Fuse Igniter

The relay system will receive a signal from the microcontroller and send a voltage to the fuses causing the charge to detonate so the parachute can be deployed. It is essential that the relay will function by the low voltage signal output by the TI MSP430.

II. Power Supply System

The power supply system will provide power to the whole unit. The device will provide the relay system with the 9V necessary to ignite the e-fuse, a 3.3V rail for the TI MSP430 microcontroller and what ever voltage the sensor system will require.

III. Sensor System

The sensor system will provide the microcontroller with an input signal that will indicate the current altitude of the system. If an analog altimeter is used to provide the microcontroller with the altitude reading the signal may need to be processed prior to entering the analog to digital converter of the microcontroller. The INA326 came highly recommended by the sponsor for this purpose. In the case of a digital altimeter, the microcontroller will communicate with the sensor via I²C or SPI.

Part Number	MPXHZ6116	MPX4115	MPL3115A2	MS5607
Analog or Digital	Analog	Analog	Digital	Digital
Output Format	Voltage	Voltage	I ² C	I ² C/SPI
Price (\$USD)	5.50	16.00	14.95	30.00
Supply Voltage (V)	4.85~5.35	4.75~5.25	1.8~3.6	1.95~3.6
Pressure Range (kPa)	20~115	15~115	50~110	1~120
Accuracy	±1.5% V	±1.5% V	±0.05 kPa	±.15 kPa
Resolution/Sensitivity	44.2 mV/kPa	46 mV/kPa	0.3 cm	0.13 / 0.084 / 0.054 / 0.036 / 0.024 mbar
Operating Temperature (°C)	-40~125	-40~125	-40~85	-40~85
Package and Dimensions	8-SSOP 30mm x 40mm x 8.25mm	6-SIP 11 mm x 7.6 mm x 4 mm	QFN 3mm x 5mm x 0.95mm	LGA 3mm x 5mm x1.1mm

Table 2.1 Altimeters Comparison Table

IV. Microcontroller

The microcontroller will take in the signal from the sensor to determine when the parachute deployment fuse should be ignited. Characteristics of the microcontroller will depend on the type of sensor system selected, if the analog sensor is to be used a high frequency and high precision ADC is desired to improve precision of the recorded values.

	G2553	F235	F5310
Frequency (MHz)	16 MHz	16 MHz	25 MHz
ADC	10-bit SAR	12-bit SAR	10-bit SAR
ADC Channels	8	8	12
Pin/Package	0DIESALE 20PDIP 20TSSOP 28TSSOP 32VQFN	64LQFP 64VQFN	48LQFP 48VQFN 64VQFN 80BGA MICROSTAR JUNIOR
MSP430 USB debugging Interface available?	YES	YES	YES

Table 2.2 MSP430 Microcontroller Section

In order to determine the optimal components to use for each subsystem, a feasibility matrix was assembled for each subsystem to compare the various solution established during a brainstorming session. Some key considerations when selecting these features were the cost of the item (due to a limited budget), complexity (due to limited time to work on the project), and package size (due to limited fabrication capabilities as well as package size).

Table 2.3 shows the estimated cost to fabricate a prototype EPD system by addressing components used in the four subsystems. The estimated cost does not address the assorted resistors and capacitors required due to their low cost and difficulty identifying exact quantities needed earlier on in the design cycle.

2.4 House of Quality

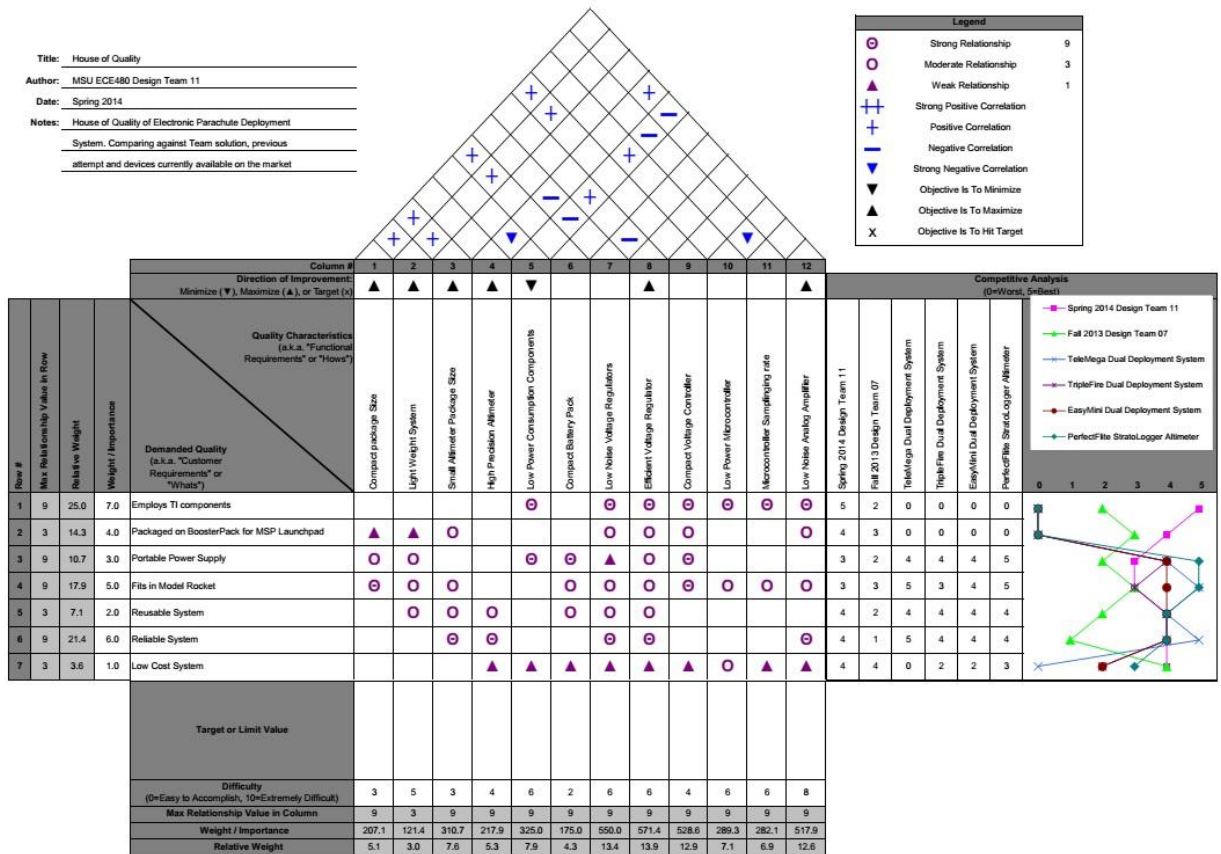


Figure 2.2: House of Quality

2.5 Estimated Prototype Cost

Part Name	Unit Price(\$USD)	Quantity	Total Cost (\$USD)
MSP430 LaunchPad	10.41	1	10.41
Analog Altimeter MPXHZ6116	5.56	1	5.56
Digital Altimeter Parallax MS5607	29.99	1	29.99
INA326	4.67	1	4.67
LM2574	1.99	1	1.99
LP2950	0.8	1	0.8
330uH Inductor	1.65	1	1.65
Relay	1.26	2	2.52
Total			55.94

Table 2.3 Estimated Prototype Cost

2.6 Gantt Chart

In an effort to efficiently allocate time and identify critical paths to complete the project the team developed a Gantt chart. Key tasks were identified early on in the project and assigned to members to complete throughout the semester. Gantt Charts allow for the user to subdivide major sections into subsections, allowing the team to initially consider all the major components of the project and then further divide each section. The chart was periodically reviewed and readjusted to ensure the project will complete by established due dates. A portion of the Gantt chart is shown in Figure 2.3, while the full Gantt chart can be found in Appendix 3.1.

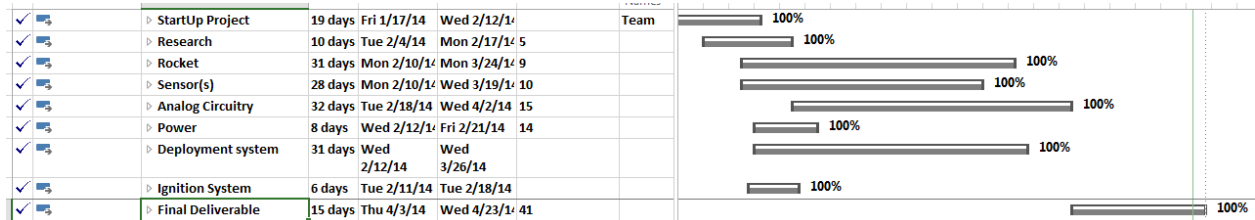


Figure 2.3 Gantt Chart

Chapter 3 - Technical Description of Work Performed

The majority of the project consisted of the researching and the initial simulating and testing of each section of the design. This deemed most important in achieving a successful final system.

3.1 Sensors

The sensors are the only inputs into our system and so they were selected carefully to achieve desired results. There were two altimeter sensors that were considered during the design process to allow a detailed comparison of the two. The final design, however, contains a single sensor.

These altimeters are pressure sensors that sense the atmospheric pressure and create an output. The atmospheric pressure changes depending on the distance from the earth's surface. It is this concept that allows absolute altitude to be determined from atmospheric pressure. There is an approximate change of 1.2kPa for every 100m of altitude above sea level. One issue the sensors avoid is varying temperature. Depending on the air temperature the atmospheric pressure will change at a fixed height. Both of the sensors have built-in temperature compensation that will negate this potential issue. After researching and testing both the analog and digital altimeters, the team chose to use the analog altimeter in the final design.

3.1.1 Digital Altimeter

The digital altimeter, MS5607, shown in Figure 3.1.1, was picked at first. The reason is that with the module chosen the accuracy can get up to 20cm and a sampling rate up to 100 samples/s. Additionally, the operational voltage range for this module is between 3.3V and 6.5V. This means the microcontroller can easily connect to the altimeter without any additional voltage regulating circuit. The communication protocols that come with this altimeter are I²C and SPI protocols. A selection pin (PS) on the module was used to switch between I²C and SPI, and I²C will be used if the PS pin is pulled high, SPI if it's low.



Figure 3.1.1 Altimeter Module MS5607

Each time the altimeter module is powered on; a recalibration process will be performed to give the correct data at later stages. It then passes in the commands to get data back from the altimeter. In this approach, the team decided to use SPI as the communication protocol. With this type of implementation, microcontrollers can easily communicate with the altimeter at a

speed up to 20MHz. The direct pressure data was not given by the altimeter directly. Instead, several computational parameters and coefficients given previously in initiation stage will be provided. As shown in Appendix 3, a number of communication cycles were needed in order to gather all necessary information required by the computation of pressure. The actual pressure generating speed will depend on the clock speed and architecture of microcontroller since there are a lot of floating points involved in the computation of the pressure. The optimized result would be at a speed up to 20 sample/s.

In an effort of making the altimeter working correctly with microcontroller, the team researched the altimeter's operation and using a SPI interface between the MCU and the altimeter. Following the datasheet and code samples given by the manufacturer of the altimeter module, the team experimented both hardware and software implementation of SPI protocols. Because the team thought it was the incorrect usage of the communication protocol that caused they could not gather the correct data. In later experiments, they discovered it was that the time delays that were programmed were not matching the expected value, which in turned caused the altimeter to stop sending data. Due to how the digital altimeter works, it needs to stay in idle while processing/generating parameters with a certain amount of time. Incorrect delay durations will cause the altimeter to stop the conversion process that being performed by altimeter.

The team continued to invest time and effort into the digital altimeter. Unfortunately, even adjusting the time delays in the program was not allowing for proper communication between the MCU and the altimeter. The team then considered the goal of the project and concluded that TI originally wanted an analog solution to this project. The digital altimeter was a point of comparison that the team suggested. Abandoning this part of the project would allow team members to work on other parts of the project, increasing the likelihood of one working solution. Therefore, the team abandoned the digital altimeter to focus on the analog solution to the project.

3.1.2 Analog Altimeter

The analog altimeter was chosen based on several considerations. It needed the ability to run off 5V or 3.3V from the power section, and its measurable altitude range had to cover the rocket's flight. The chip needed to be large enough for soldering by hand, ideally a surface mount package, and needed to include temperature compensation. After extensive research the team chose Freescale Semiconductor's MPXHZ6116. This sensor continuously outputs a varying DC level proportional to the atmospheric pressure, allowing it to be very fast. Its speed is only limited by the rate at which it is sampled from the MCU. This sensor, however, only changes its output by 44.2mV per kPa, with an accuracy of 1.5%, making the signal change fairly small. The final design has the signal being conditioned through analog circuitry before being sent to the MCU to enhance the amount of change. The altimeter's altitude range is 20kPa to 115kPa, with a 0.399V to 4.645V output, thus satisfying the initial considerations. Also, it runs off a 5V supply, contains temperature compensation, and is a Shrink Small Outline Package (SSOP). Equation 3.1.1 calculates the output based on absolute pressure, and Figure 3.1.2 below shows the output vs. absolute pressure for the analog altimeter.

$$V_{out} = V_s * (0.008938 * P - 0.09895) \pm (1.5 * TM * V_s * 0.008938) \text{ (Volts)}$$

Equation 3.1.1 Calculation of Analog Altimeter Output
[TM=1, P=pressure (kPa), Vs=source voltage (5V)]

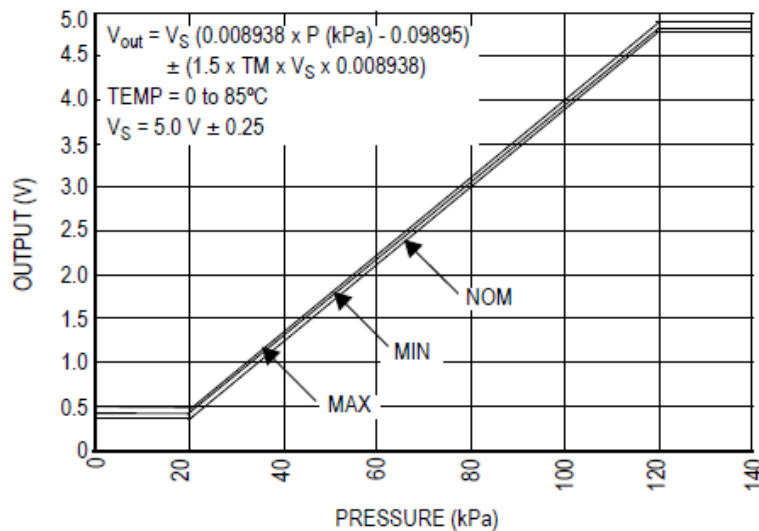


Figure 3.1.2 Analog Altimeter Output vs. Absolute Pressure

As one can see, the voltage output of the altimeter increases with an increase in pressure. To test the altimeter a straw was used to blow into the sensor, causing the pressure to increase. Figure 3.1.3 below shows the output before, to the left, and during, to the right, the straw test.

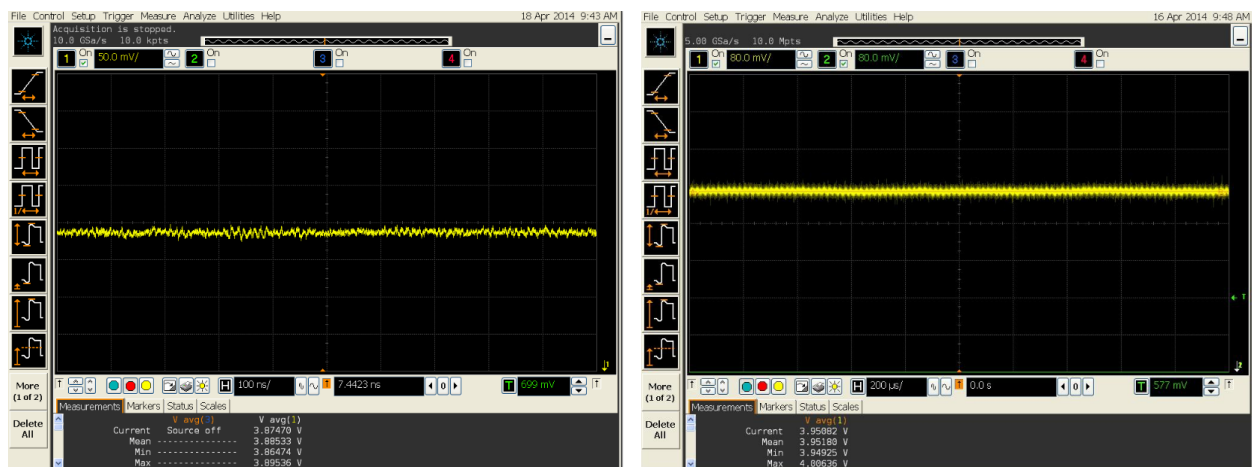


Figure 3.1.3 Analog Altimeter Output Before and During Blow Test

The average voltage of the output before the test was approximately 3.87V while the output of the blow test showed 3.95V. However, keep in mind as altitude increases, the absolute pressure decreases.

Michigan is typically around 1000ft above sea level, and the model rocket is built for a 1000ft launch. The initial design had a larger altitude range than what is needed for Michigan, allowing it to be used in other regions. Using the output from the analog altimeter, as stated

before, does not provide much change in V/kPa, making it difficult for the MCU to distinguish between changing voltage levels. To increase the change, the output needed amplification.

3.2 Analog Amplifier

The analog altimeter's output needed to run through some sort of amplifier in order to increase its change in V/kPa. Multiple considerations were taken into account when the amplifier was being chosen and when the circuit was being designed. First, the output of the amplifier, which leads to the MCU, needed to satisfy the voltage range of the MCU's ADC while simultaneously satisfying the desired altitude range of the rocket. Also, some sort of amplification with inversion or a DC offset was needed to lower or invert its output. The final design uses Texas Instrument's INA326 instrumentation amplifier. However, this was not the only amplifier considered.

3.2.1 Operational Amplifier

The team originally chose to use Texas Instrument's OPA335 operational amplifier. The idea was to use this device as an inverting amplifier to the altimeter signal. The signal would enter the op amp, be inverted, have a DC offset added to it and then be amplified. The team tested this idea in TINA-TI. The input into the circuit was a voltage generator that acted as the input coming from the analog altimeter. Using Equation 3.2.1 the absolute pressure at ground level in Michigan (1000ft above sea level) was calculated to be 97.716kPa.

$$P = 101325 * (1 - 2.25577 * (10^{-5}) * \frac{h}{3.2808})^{5.25588} \text{ (Pa)}$$

Equation 3.2.1 Calculation of Absolute Pressure [P=pressure, h=height (ft)]

The max height at a 1000ft launch, or 2000 ft above sea level, is about 94.213kPa. Again, the system at the time was designed to be used at areas other than Michigan so a larger voltage input was used. The pressure around 13,000ft above sea level is about 61.9kPa. The voltage input into the amplifier was calculated using Equation 3.1.1 at both 97.716kPa and 61.9kPa. This gave a voltage varying from 3.94V to about 2.3V. Figure 3.2.1 shows the circuit built in TINA-TI along with its component values, and the transient response can be seen in Figure 3.2.2 below.

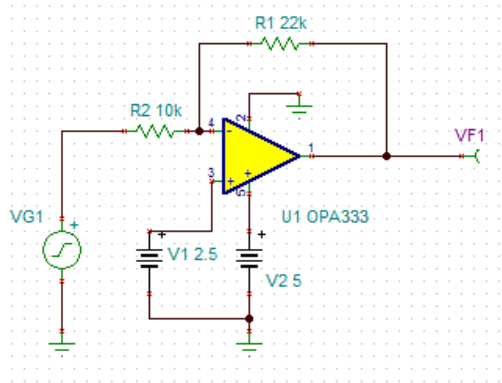


Figure 3.2.1 OPA Schematic Built in TINA-TI

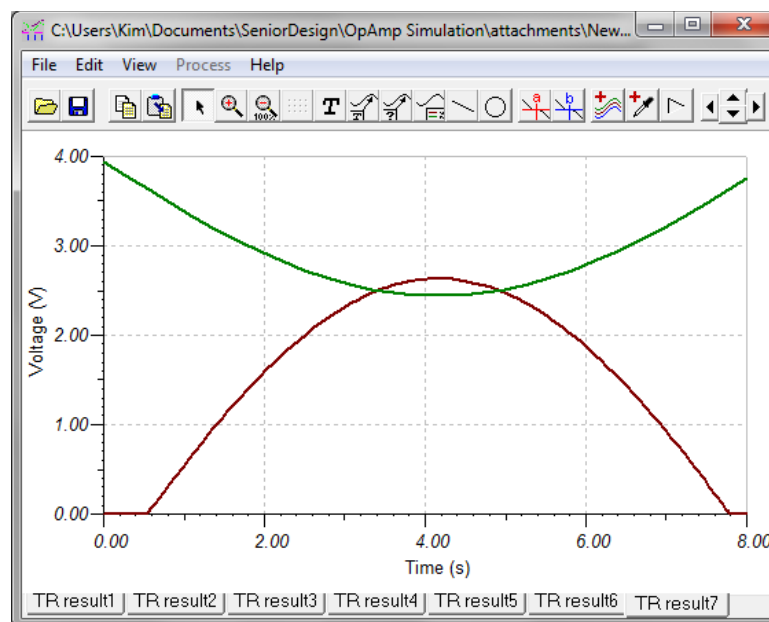


Figure 3.2.2 OPA Schematic's Transient Analysis

The green line represents the input into the OPA and the red line is the output that goes into the ADC. The MCU selected at the time could take in as much as 3.3V from the altimeter to be converted using the ADC. As can be seen from the graph the output ranges from 0V to 2.6V, satisfying the maximum criteria but clipping for the first few seconds. Also, this op amp outputs noise due to its feedback loop at a gain of 1. This forces the input into the ADC to be less accurate. The output going into the MCU should be as clean and noise free as an amplifier can get. The group was able to further research and found a better option.

3.2.2 Instrumentation Amplifier

The team found Texas Instrument's precision instrumentation amplifier, INA326, for the final design. This option has very low DC errors and noise. The instrumentation amplifier is essentially a circuit component consisting of three operational amplifiers. Two of the op amps receive a single external input. The other input pin is a resistor that connects the two. This

system creates a signal that is the difference between the two inputs. The third op amp is used to buffer the signal after a gain stage.

The INA is first used to attenuate the altimeter signal. The voltage of lowest point one would expect at a launch is placed on the non-inverting input of the INA. The signal of the altimeter itself is placed on the inverting input. The difference of the altimeter voltage from the minimum altitude's voltage moves the range of the altimeter so that approximately 0V on the output is the lowest altitude possible. The signal then gets amplified, and the final output thus increases as the altimeter output decreases.

The signal going into the non-inverting input of the INA was chosen to be a value just above the 3.94V so the difference is not negative. The 5V coming from the power section and a voltage divider are used to input around 4.257V (V^+) into the non-inverting pin. The gain was chosen and designed to be 2. Also, an RC filter on the output of the INA was added to the design to reduce noise and act as an anti-aliasing filter that inputs into the ADC on the microcontroller. Equation 3.2.2 calculates the expected output coming from the INA circuit.

$$V_{out} = [(V^+) - (V^-)] * (gain)$$

Equation 3.2.2 Calculation of INA output

The INA circuit was built and tested in TINA-TI to see the response. Figure 3.2.3 shows the circuit, and Figure 3.2.4 shows its transient response.

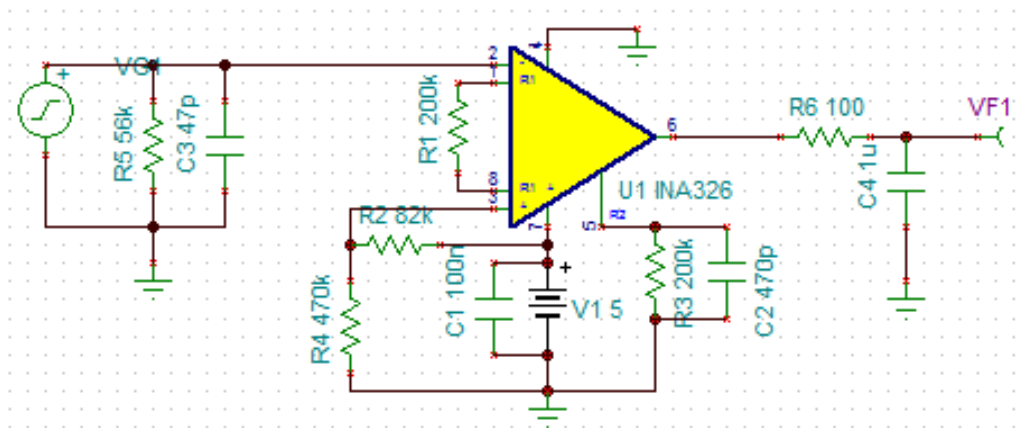


Figure 3.2.3 INA Schematic with Gain of 2

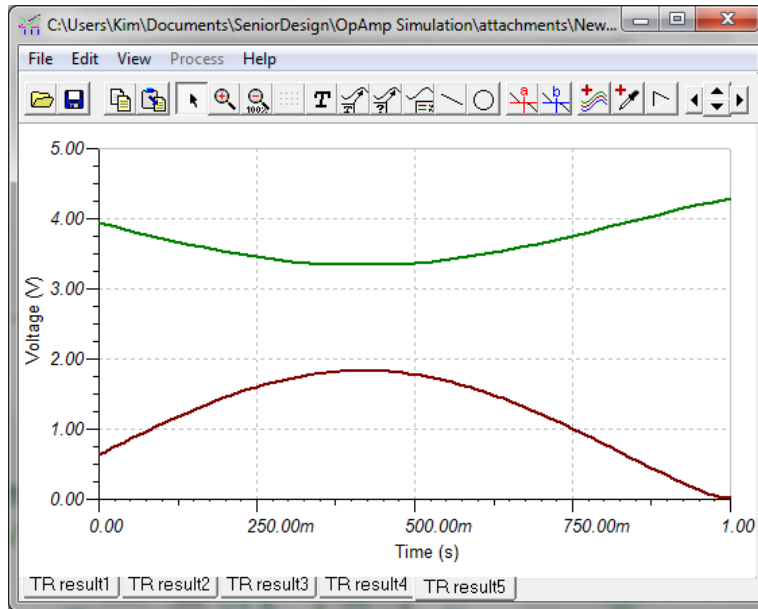


Figure 3.2.4 INA Schematic's Transient Response with Gain of 2

The green input line, again, represents the input of the altimeter. This time the simulation was ran to see how the INA would respond to an altimeter range closer to what is expected for this project's rocket launch. It ranges from the expected 3.94V down to 3.3V or from 1000ft above sea level, the starting launch, up to about 5000ft above sea level. The red line shows that the output of the INA ranges from 600mV to 1.8V. The MCU had since been changed and had a 2.5V limit coming from the ADC. This simulation verified the INA will not clip with a 1000ft launch from Michigan. As the project continued on more changes were made. The team decided to increase the gain for better accuracy, which decreased the useable location range for the overall system. The gain was changed to be around 4.878 by switching out one of the resistor values. Figure 3.2.5 shows the new circuit along with its transient response, in Figure 3.2.6.

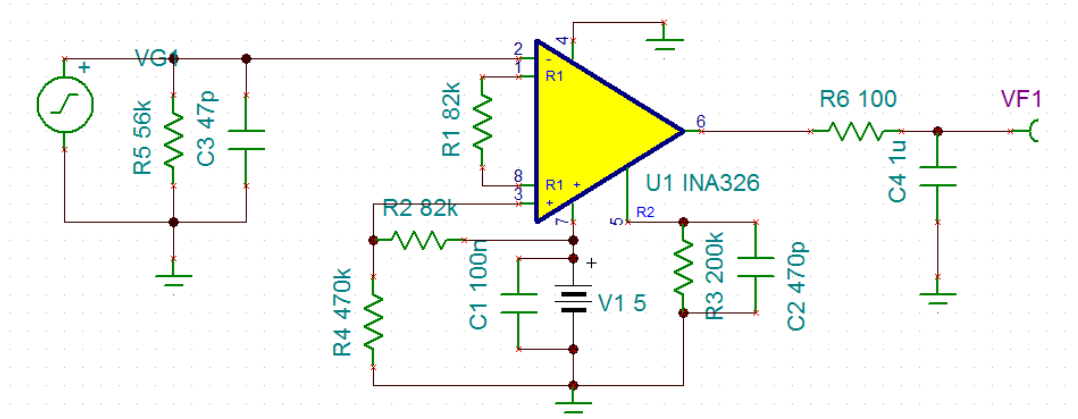


Figure 3.2.5 INA Schematic with Gain of 4.88

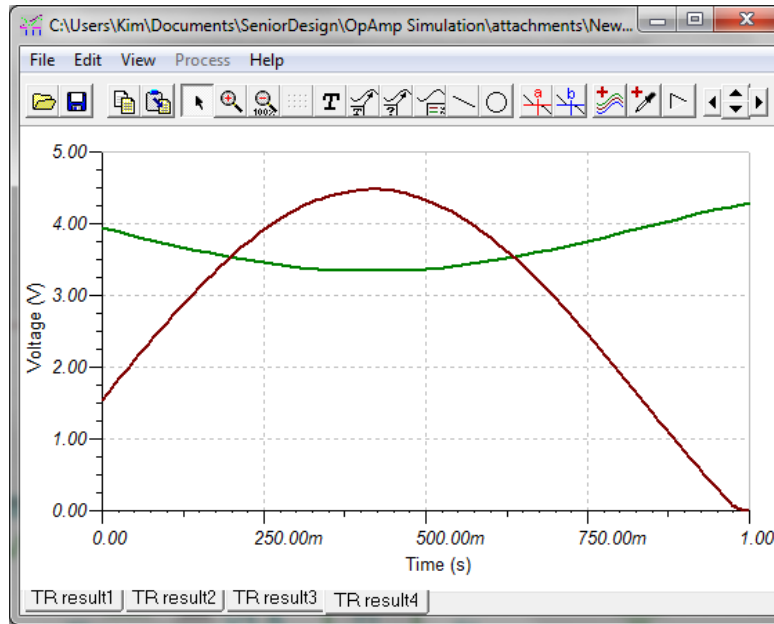


Figure 3.2.6 INA Schematic's Transient Response with a Gain of 4.88

The input was kept the same. The start of the launch is around 3.94V, as stated before, and at the max height of the launch it is at about 3.78V. The red output line shows that the output of the INA ranges from 1.55V (at the 3.94V input) to about 2.3V (at the 3.78V input), not exceeding the 2.5V limit coming from the ADC, thus verifying the INA will not clip with a 1000ft launch from Michigan.

The INA and altimeter were next bread boarded to verify the parts worked as expected. The proper components were all in place and the output was correct but with minor noise. This was due to the INA being very sensitive to parasitic capacitance when bread boarding. In order to reduce the amount of noise, the resistor going across the inverting and non-inverting pins was soldered onto the actual pins. Also, the resistor and capacitor that constitute the actual feedback path within the chip needed to be directly soldered onto their pin as well. With these changes, the output of the INA looked a lot better with reduced noise, confirming that the chip would perform as expected when on the PCB. Figure 3.2.7 shows the relationship between the output of the altimeter, green line, and output of the INA, orange line.

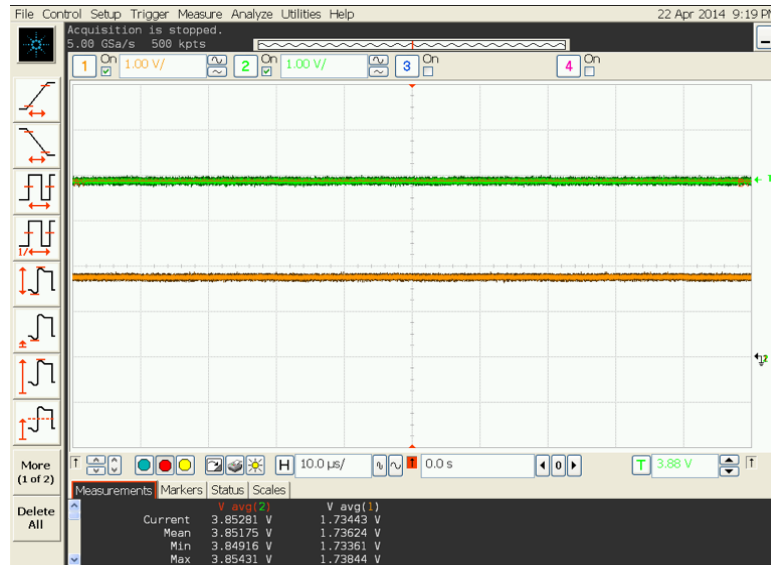


Figure 3.2.7 Analog Altimeter and INA Output

The altimeter output shows 3.85V and the INA displays 1.73V as expected.

3.3 Microcontroller

3.3.1 Hardware

The microcontroller for the EPD system is the MSP430G2553. This microcontroller, along with the MSP430F235, were considered for the initial design of the project. The original criteria for the microcontroller was to have a solderable package, a high-resolution analog-to-digital converter (ADC), and be compatible with the MSP430 LaunchPad. The MSP430F235 was considered because it has a 12-bit ADC. Using a 12-bit microcontroller using the internal 2.5V reference would give a resolution of 610.5uV. The MSP430G2553 has a 10-bit ADC, which has a resolution of 2.44mV, which is four times less resolution. Once ordered and examined, the MSP430F235 was deemed inappropriate for the design. The MSP430F235 has 64 pins with a pin spacing of .5mm and pin size of 27mm. This is very difficult to solder by hand, which was one of the criteria of circuit pieces required for the project. In addition to the close spacing of the pins, the package required the user to have a breakout board and additional hardware to program the microcontroller. This led to finding a different microcontroller, the MSP430F2013.

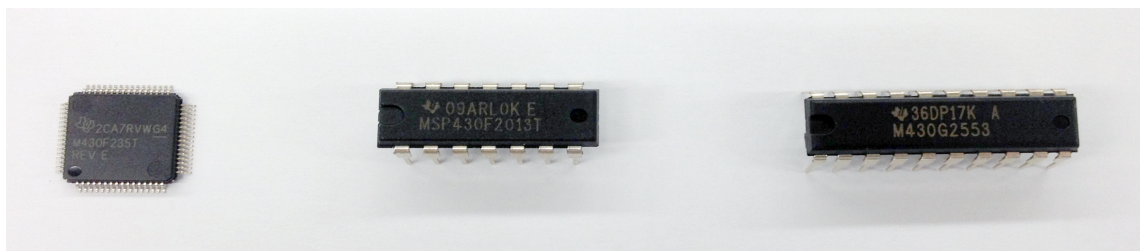


Figure 3.3.1 The three MSP430s considered for this project: MSP430G235, MSP430F2013, and MSP430G2553 (left to right)

The MSP430F2013 seemed to be a promising microcontroller for the project. The packaging for the MSP430F2013 that was used was a 14-pin DIP package, which fit inside the LaunchPad. This allowed for easy testing and programming. The rationale behind using the MSP430F2013 was to take advantage of its 16-bit ADC, which has a resolution of 38.15 μ V. After testing the ADC and consulting the datasheet, it was determined that the microcontroller could not be used because of the way the ADC was implemented on the device. The ADC for the MSP430F2013 is a Sigma-Delta ADC, which uses differential inputs and a maximum reference voltage of 1.5V. The implementation of this ADC puts the maximum input voltage at $\pm V_{ref}/2$. Half of this voltage range would be lost because this project would operate in the unipolar mode, going from 0V to .75V. This presents a difficulty as the output of the INA326 at ambient pressure in East Lansing translates to a value between .6 V and .7 V. The change in voltage for a rocket launch of 1000ft would be approximately 380.6mV in change, which would definitely be out of its usable range. With these issues, the team decided to abandon the MSP430F2013 and use the MSP430G2553.

The MSP430G2553 is the microcontroller used in the final design of the project. It has a 20-pin package, which also fits in the MSP430 LaunchPad. It uses a 10-bit ADC, which was found to be adequate for detecting proper altitudes for parachute deployment when tested. The input range of the ADC goes from 0V to V_{ref} . Although the design implementation can use an external reference that can be as high as V_{cc} , the final design uses the built-in 2.5V reference voltage. In addition to having manageable packaging and a usable range for the ADC, the MSP430G2553 was familiar to all team members, which made programming and debugging the device easier than the previous design choices.

3.3.2 Software

The software for the EPD system is designed to detect the peak of the rocket flight and send a signal to deploy parachutes at appropriate altitudes during the descent. The criteria for the code was that it had to correctly detect the launch of the rocket, detect the peak of the rocket flight, detect or calculate the percentage of the total height above ground, and send out two separate signals that would trigger the deployment charges that release the parachutes. In order to do the detection, the code would have to accurately read input data from either the analog altimeter, the digital altimeter, or both.

In the original design of the project, it was proposed to compare the analog and digital altimeters side by side to see which device worked the best with the rest of the EPD system. The analog altimeter would need to be filtered and presumably gained, the process in which was described previously. The digital altimeter would communicate directly to the microcontroller through I2C or SPI interface. The digital altimeter, as previously described, has calibration data that must be read in order to calculate the pressure. Therefore, the code needed to read multiple bytes of calibration data before polling for data.

The team first decided to communicate to the digital altimeter using a SPI interface. The SPI interface was chosen as SPI does not require extra components to implement and can operate at speeds up to 20MHz. The digital altimeter operates at a maximum of 400kHz with I2C. The team invested a significant amount of time to try to properly implement the system. There was code developed that was capable of communicating with the digital altimeter and

was able to receive some data. The data for calibration that was transmitted, however, was not similar to the expected values.

Debugging this code and searching for answers to the coding problems was consuming too much time. Instead of continuing to try to make the digital altimeter work, the team changed its focus to creating and debugging a code that only received inputs from the analog altimeter. This would increase the likelihood of having a working project with code, instead of having two incomplete codes at the project end date.

The final code for the EPD system is designed to work on the MSP430G2553 and an analog altimeter. The ADC of the microcontroller reads the analog altimeter's signal. The code takes the first value it reads and considers this to be the base altitude. It then waits for a significant change in the altitude to detect launch. Once launched, it continually polls for the highest altitude. Once the microcontroller reads a significant drop in the altitude, it begins to compare the current altitude to a percentage of the apogee of the flight. Once it has detected a true decrease in altitude, the code sends out a signal to deploy the drogue parachute. It sends a signal to deploy the main parachute at 50% of the total flight height.

3.4 Parachute Deployment Fuse Igniter

A successful parachute deployment all starts from a properly assembled rocket. The larger model rocket (Madcow Torrent) will contain two parachutes and will be controlled by the tested EPD system. The figure below shows the different rocket sections within this larger rocket. In order to launch the parachute the EPD system requires the system to supply 2A of current to a fuse. The fuse would then light a match to half a teaspoon of pyrodex. This creates a force large enough to separate the nose cone from the body to deploy the parachute.

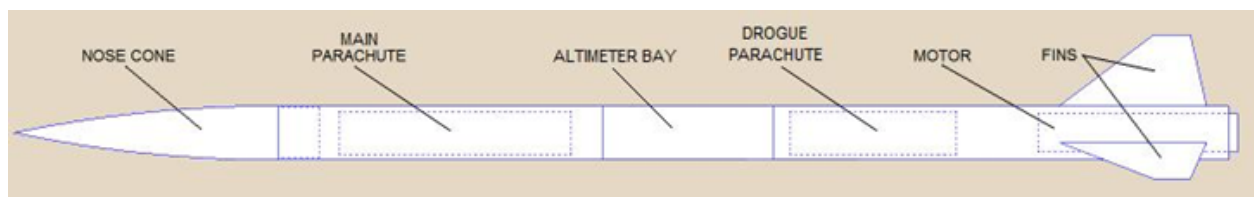


Figure 3.4.1 Sections of Model Rocket

The team initially spent a significant amount of time trying to determine a solution to supplying the proper voltage to the system. The first attempt the team tried was to use two n-MOSFETs, a Schottky diode, and a capacitor to charge the system. The MOSFETs would become saturated through outputting a high signal from the microcontroller on independent outputs. This was meant to provide enough time for the capacitor to charge from the battery, return the first MOSFET to the cutoff state, and saturate the second MOSFET to burn the fuse.

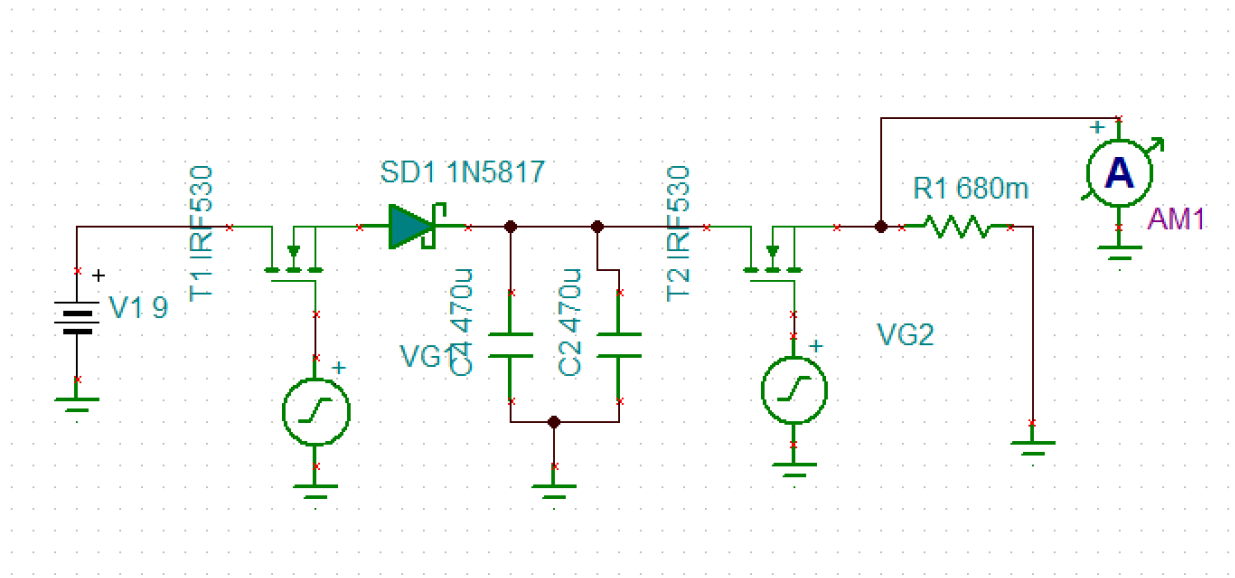


Figure 3.4.2 Initial Igniter Circuit

Although the TI-TINA claimed the system should be able to implement the desired 2 amp output, when implementing this configuration on a breadboard we did not obtain the same desired results. The team also decided to order relays as an alternative to using MOSFETs trying to achieve the same implementation. The relay was picked to ensure the coil was energized based on the MCU output voltage. The team chose to use a Omron G6L-1F which required a voltage of only 3V. The relay has coil resistance at 50 ohms which means the rated current using ohms law is 60mA as provided in the data sheet. The maximum voltage intake for the relay is 150% of the maximum rated voltage, which at 3V, is 4.5V. Also the minimum required voltage for operation is 75% of the rated voltage, so the total operation range is between 2.25V to 4.5V. The microcontroller can output $VCC+0.3V$ and the input voltage (VCC) is at 3.3V; therefore the 3.6V output is well within tolerance of the required voltage to energize the coil. The team also chose the relay because it is in a normally open orientation. This allows the connection between the battery and the ignition to remain open until the microcontroller energizes the relay coil. This is important because the team does not want to connect the battery to the system and have an automatic short to the battery accidentally triggering the fuses while the person is setting up the system.

■ Coil Ratings

Item	Voltage Rating		
Rated voltage	3 VDC	4.5 VDC	5 VDC
Rated current	60.0 mA	40.0 mA	36.0 mA
Coil resistance	50.0 Ω	112.5 Ω	139.0 Ω
Pick-up voltage	75% max. of rated voltage		
Dropout voltage	10% min. of rated voltage		
Maximum voltage	150% of rated voltage		
Power consumption	Approx. 180 mW		

Figure 3.4.3 Relay Coil Ratings

Outputs, Ports Px

over recommended ranges of supply voltage and operating free-air temperature (unless otherwise noted)

PARAMETER	TEST CONDITIONS	V _{CC}	MIN	TYP	MAX	UNIT
V _{OH}	High-level output voltage	I _{I(OHmax)} = -6 mA ⁽¹⁾	3 V	V _{CC} - 0.3		V
V _{OL}	Low-level output voltage	I _{I(OLmax)} = 6 mA ⁽¹⁾	3 V	V _{SS} + 0.3		V

(1) The maximum total current, I_{I(OHmax)} and I_{I(OLmax)}, for all outputs combined should not exceed ±48 mA to hold the maximum voltage drop specified.

Figure 3.4.4 MCU Output Voltage

The team then tried implementing the same circuit with the relays instead of the MOSFET. By closing one relay the team could observe the capacitor charging to the desired voltage and watched the voltage drop very slowly after reopening the same relay. The issue we encountered was when closing the second relay; the charge dropped as expected, but did not light the fuse. The team tried adding additional capacitors, up to 1mF, in parallel to increase the amount of stored charge, but regardless, the team could not provide enough charge to ignite the fuse.

Although these two methods would have greatly benefited the safety of the system, the team finally implemented a solution that used only one relay, a 5Ω resistor, and the igniter. This solution uses a relay to prevent the system from false firing during set up and a 5Ω resistance to pull the necessary current from the 9V battery to ignite the fuse. The team was able to breadboard the fuse and then test the final implementation on the PCB. Figure 3.4.5 uses the relay1 as an output reference to represent the fuse.

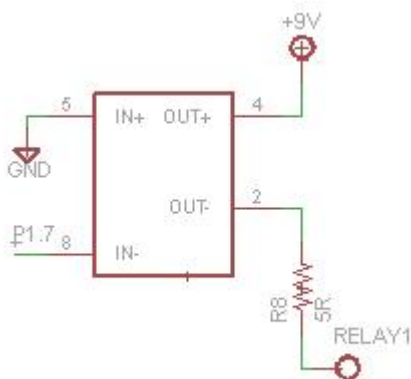


Figure 3.4.5 Final Fuse Ignition System

3.5 Power System

A power system is needed in the EPD system to convert the systems power source to the needed voltage potentials required by the active components. This power system was developed after the key active components in the design were chosen to ensure complete compatibility with the rest of the system. The most significant challenge of this system design was balancing efficiency, physical size, and cost. With the final power system design the team has developed a system with a good balance of these design criteria.

The power system of the EPD must meet the voltage and current needs of the other components within the system. The system needs three main outputs to the other components

of the system. It must provide a 5V output to both the analog altimeter and the signal condition amplifier. The system also must output 3.3V for the microcontroller. The last output is a short 2A burst output to power the igniters.

The power source chosen for this EDP system is a simple 9V battery. This is the standard power device for commercial rocket altimeters and provides a large amount of charge in a small physical space. Other options considered were a Lithium ion coin cell battery. The 9V battery holds roughly five times the amount of stored energy

Once the power source was determined some power conversion was needed to change the 9V input to a 5V and 3.3V output. There are a few topologies that would allow this step-down conversion. The system could use a linear voltage regulator, a charge pump or a buck (step-down) converter. These power systems each are unique and offer different benefits.

A linear voltage regulator is typically quite cheap and usually requires two decoupling capacitors and the voltage regulator itself to operate. These devices can take a voltage at a high potential and output a lower voltage. This is done by dissipating excess voltage as heat, and therefore is quite inefficient.

A charge pump system operates by using a single integrated circuit that contains a series of switches and capacitors. Within the IC the input voltage enters and then is passed between capacitors using switches. These capacitors are to be sized in such a way that the voltage can be increased or reduced at the output of the device. These devices are more efficient than linear voltage regulators. The issue with charge pumps is that all of your power must go through the IC, which has a fixed amount of power it can handle. This usually limits the output current of charge pumps to the milliamp range.

Finally there is the buck converter topology. This topology is based around using an inductor as a storage device. Figure 3.5.1 below shows the basic design of the power system.

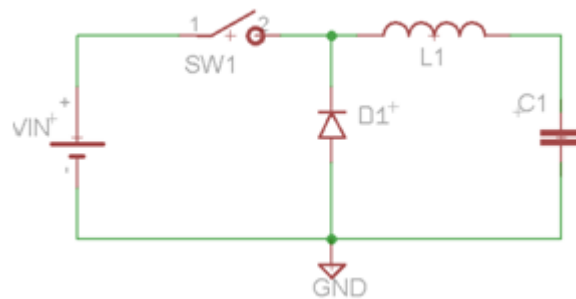


Figure 3.5.1 Simplified Buck Converter

When the switch in the buck converter is closed, the inductor coil will charge up according to the equation $v(t) = L(di/dt)$. After a certain time, the switch will open. The inductor will try and dissipate its stored energy out to the capacitor and the load that would be hooked up in parallel to this capacitor. This along with the fact that the switch is open will cause the node between the diode and the inductor to be at a negative potential compared to ground. This will forward bias the diode and allow a small amount of charge to flow back into the inductor. By carefully regulating when the switch is opened and closed and by choosing the inductor you will be able to store energy in the inductor and release it slowly during discharge cycles. This allows the step-down of voltages with good efficiency.

The team initially chose to try and use a step-down (buck) converter to do this power conversion. This converter would be quite efficient, but could be a bit more difficult to implement. In buck converter system you can buy an integrated control that will operate the switch and it will have some switching regulation control method. Also buck-converter integrated circuits may also move a switching transistor or the diode on to the chip, so less external parts are needed.

The team initially looked at trying to use a two-output buck converter IC, for example the TPS54386 from TI. This single chip along with external components would then change our 9V input to a 5V and 3.3V output. These dual type buck converter chips from TI required many external components and were quite complex. Using a dual buck converter system to get both output voltages would require more physical area and could increase the cost of the system. Another potential hazard is having the two inductors from the different supply rails couple in some way causing instability.

It was then determined to attempt to just use the buck converter system to output the 5V rail. This conversion from 9V to 5V could then be made very efficient. This would be beneficial to the overall efficiency because the two active devices, the analog altimeter and its instrumentation amplifier, on average require more current than the microcontroller running on the 3.3V rail.

With the 9V to 5V conversion in mind we set out to find a buck-converter IC that will be efficient, require few external components, and be fairly inexpensive. Our first choice was Texas Instrument's TPS560200. This converter met all the previous requirements, however when we received the chips the team realized that a new criterion was needed. These ICs were very small and would be difficult to solder by hand. So the team started searching for another buck converter IC with a better package. A new converter that looked promising was then found. This was the TPS62160 from Texas Instruments. This part worked well initially, but when more thorough testing was done a potential hazard was found. When testing this buck converter IC the team added an increasing amount of output capacitance to the output of the system. This was done while observing the effect of the total output capacitance on the output voltage ripple. More output capacitance would result in less output ripple which is desired in the EPD application. While increasing the output capacitance, the team noticed that the IC would go unstable and into thermal shutdown with an output capacitance greater than 220nF. At the point of instability the output ripple was quite small. It was decided not to use this IC knowing it has this stability issue. A third buck converter IC, Texas Instrument's LM2574, was then tested thoroughly and no hazards were found. This was found to be very robust, it required very few external components, and was the cheapest solution we had tried, so this is the chip chosen to be in our design.

With the buck converter then 5V supply was created, but there still needed to be a 3.3V rail created. Applying the 5V rail into the linear voltage regulator completed this objective. This method is not as efficient as the buck converter, but is very cheap and requires a minimal number of parts. This method does decrease the efficiency of the system, but the 3.3V power rail only powers the MSP430. The MSP430 requires little current to operate so the power dissipation in the linear voltage regulator is minimized. Initially the part used for this voltage regulator was the LM3940 in a T220 package. This then was changed in the final circuit revision to a LP2950, due to size and power needs. The original LM3940 was quite bulky and had the ability to provide a large amount of output current, which in our application was unused. The

LP2950 was nearly identical in functionality to the original voltage regulator, except that it was smaller and had a limit of 100mA of output current. For the EPD system application this new voltage regulator will work.

This designed power system provides a good mix of efficiency and cost for the EPD system. With the way the system is designed the power source is also flexible to allow voltages between 7 and 40 volts to be used. This feature was never sought after in the design, but this does allow the system to provide good voltage levels even when batteries lose their voltage potential slightly before dying. More specifically the system could also be used with the 12V A23 type battery that is fairly common. This power system has been in place through violent rocket launch testing and still is holding up to the power needs of the EPD system

3.6 Printed Circuit Board

The PCB is an essential part to the project as it is necessary to fit all the components for the EPD system inside the altimeter bay. To complete the PCB all the necessary components needed to be chosen first for each of the previously described systems. These systems include power, altimeter, ports used on the MCU, the INA, and the ignition system. A PCB includes both a schematic and layout. The schematic enables an easy visualization of the system and how the system is wired together. After the schematic is complete, the layout is made to show how the parts are placed on the fabricated board and place the traces between the components. The team completed the first three board revisions in PCB Artist and completed the final version in Eagle.

The first step of the PCB is to build the schematic. In PCB Artist the team was required to build several components in the library to properly populate the parts library for the projects PCB. Some components that were not included in the parts library include the altimeter, relay, and the instrumentation amplifier. Figure 3.6.1 below represents how the students created both the schematic symbol and PCB symbol for the SPST relay.

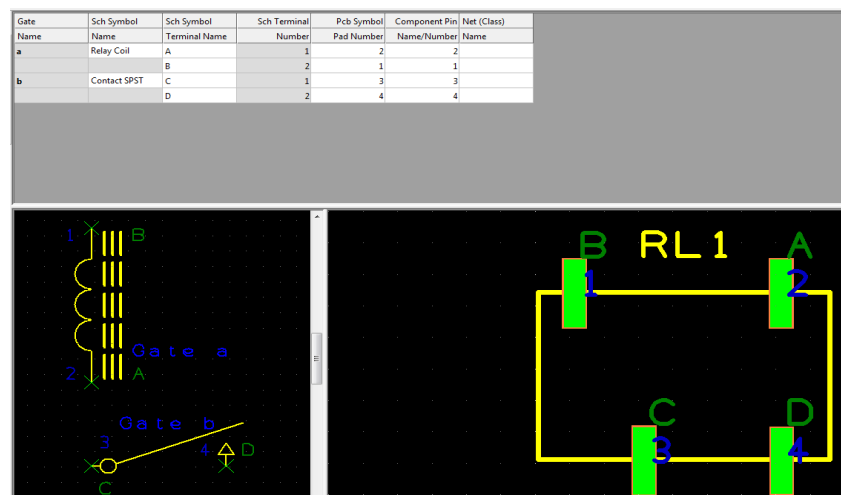


Figure 3.6.1 Relay Schematic/Symbol

After all of the parts were populated into the schematic, the team wired the components into a complete system. As components were changed in the project, it was important to update values on the schematic to ensure continuity for the team understanding what parts were on the schematic. The image below represents the final schematic used in Eagle. The team switched to Eagle for the last revision to include a pre-made component that was downloaded on the TI website for the MCU and graphic art on the PCB.

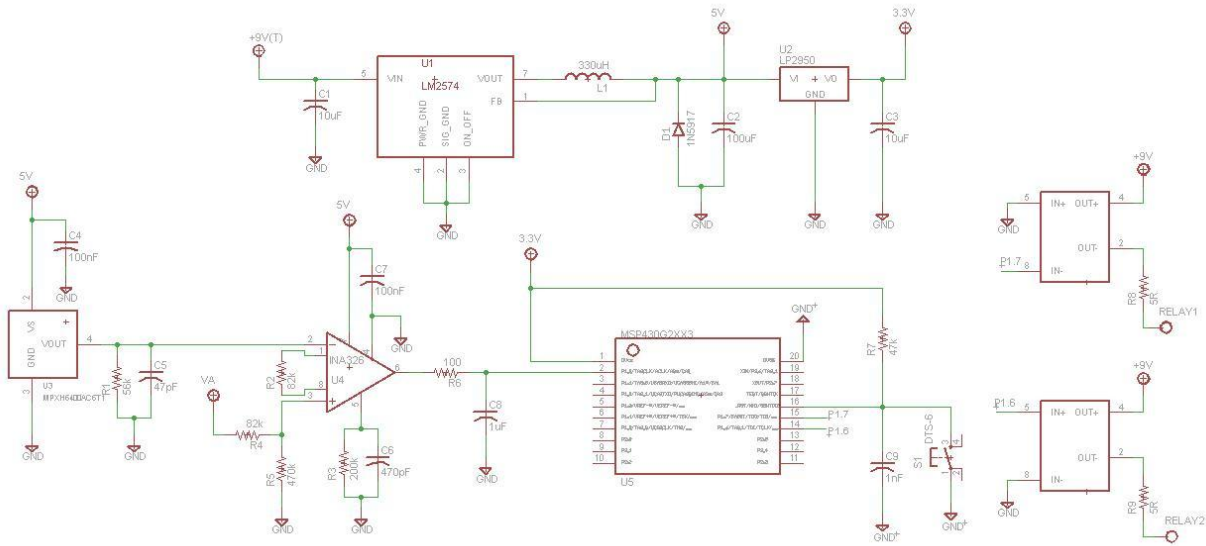


Figure 3.6.2 Final Schematic

After the schematic was complete, the next step was to layout the components and route the layout. The layout went through several revisions throughout the duration of the semester. The first version the team completed was reviewed with the sponsor and was completed while doing final tests on the breadboard.

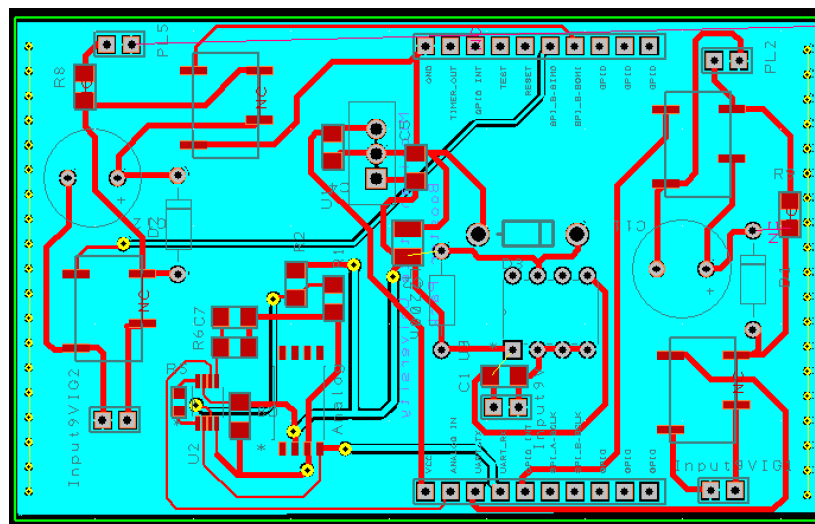


Figure 3.6.3 Initial Layout

The goal of the next layout was to modify the components from the first revision after testing the components. One major change to the layout was switching the igniter circuit from using multiple relays to a single relay circuit. The second revision the team completed was sent to Advanced Circuit and initially purchased as a barebones. This allowed the team to confirm the PCB properly functioned and purchase a cheaper PCB without silkscreen and solder mask.

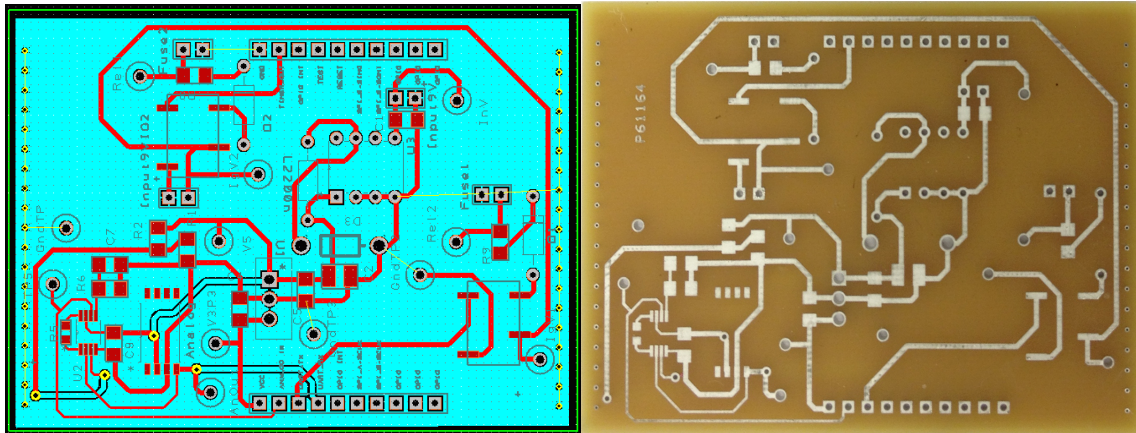


Figure 3.6.4 Layout 2 and Barebones PCB

While the barebones was being fabricated, the team then spent time trying to reorganize the layout to keep all the traces on one layer. The team modified the components to place the power components on the top of the board and the main signal path including the altimeter and INA to the center of the BoosterPack. This allowed the team to be able to remove the vias connecting traces between the top and bottom side of the layout. Also by stitching a top and bottom ground plane to the PCB the team was able to remove vias connecting components directly to the PCB and instead the ground layer would automatically add a thermal relief pad connecting the pad to the ground copper layers.

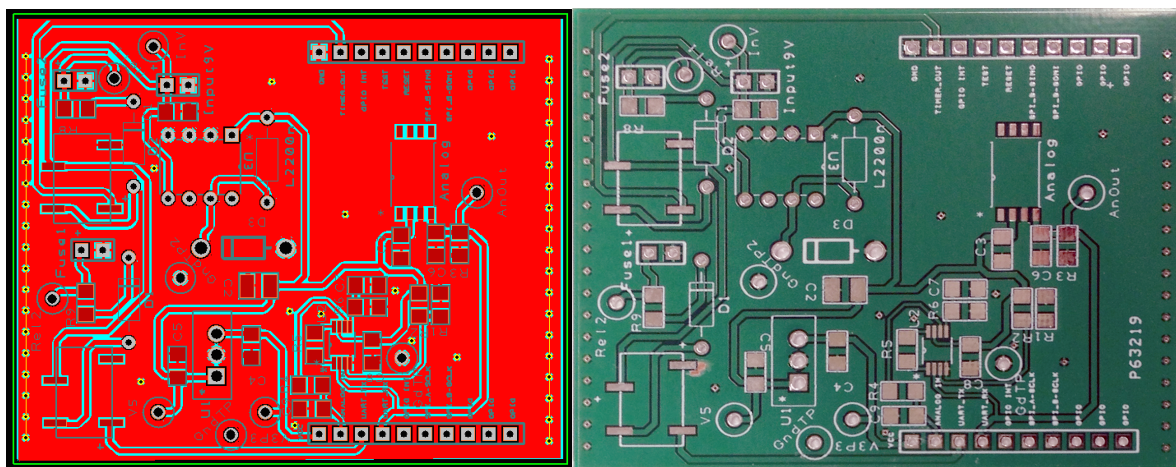


Figure 3.6.5 Revision 3 of the PCB

The fourth and final revision to the PCB created a monolithic design by setting the MSP430 onto the BoosterPack. This PCB allows for the size of the system to both reduce in

height and weight. By creating a single board design the EPD system appears as a device that is ready for a rocket hobbyist to purchase and implement. The team also added silkscreen to the PCB that included a TI logo to represent our sponsor and the university and team number. The team will be receiving the final system to populate the components the day before design day; therefore there will not be an included PCB in the report.

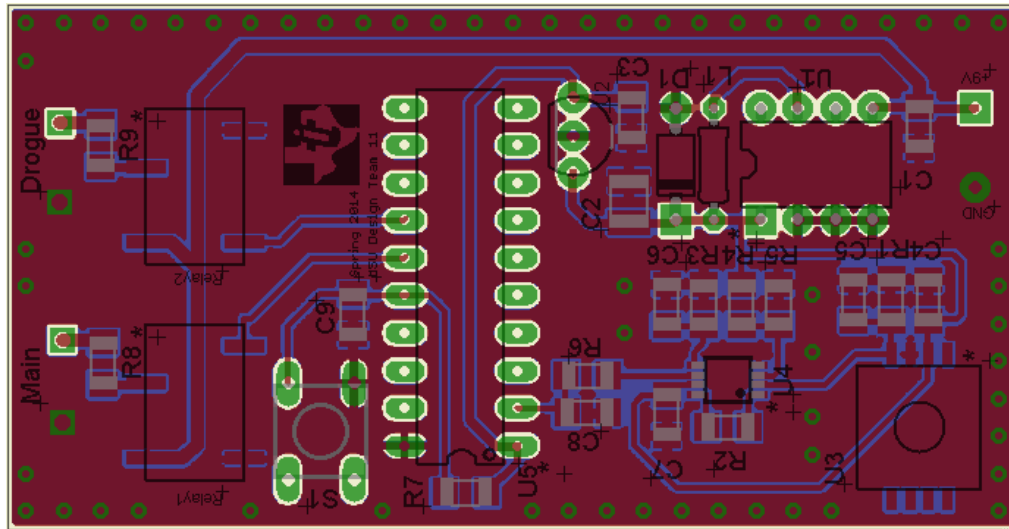


Figure 3.6.6 Final PCB Layout

Chapter 4 - Test Data with Proof of Functional Design

The final test of the whole EPD system came from launching. Other minor tests were conducted with the system as a whole before the rocket launch to ensure proper function in case something happened during the launch.

4.1 Straw Test

The first complete system check the team performed with a bread-boarded system was to perform a straw test. The team initially decided using a straw was a simple way to cover the altimeter sensor and either blow or suck air to simulate pressure increasing or decreasing. As the rocket climbs in altitude the air pressure drops and therefore the team was required to initially suck air into the straw and when the straw was released the pressure would begin to recalibrate back to its surroundings. The expected result of the breadboard would be to have the students decrease the air pressure and have the LED's flicker on consecutively simulating the drogue chute and main chute releasing.

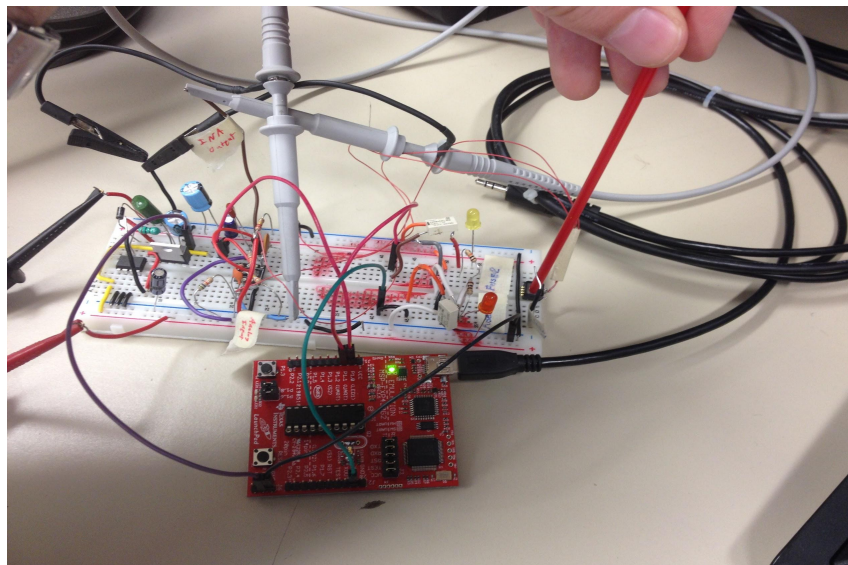


Figure 4.1.1 Breadboard Straw Test

The team is able to debug this test is operating properly through using the microcontroller in debug mode in code composer studio. While the program is running, the team set breakpoints on the variables set to display the initial room pressure from the ADC, which ranges from a scale of 0 to 1023. The team also had the code update the variables for the peak altitude, and the value the rocket is releasing the main chute. The main chute should be released halfway between the peak and the base altitude. This can effectively be seen in the image below.

Expression	Type	Value	Address	Expression	Type	Value	Address
0x> basevolt	long	650 (Decimal)	0x03C6	0x> basevolt	long	650 (Decimal)	0x03C6
0x> vmax	long	650	0x03D2	0x> vmax	long	1008	0x03D2
0x> voltRange	long	0	0x03C2	0x> voltRange	long	358	0x03C2
0x> drogueDeploy	long	650	0x03D6	0x> drogueDeploy	long	837	0x03D6
0x> mainDeploy	long	650	0x03DA	0x> mainDeploy	long	773	0x03DA
0x> currvolt	long	650	0x03CA	0x> currvolt	long	773	0x03CA

Figure 4.1.2 Start and Stop Straw Test Values

After validating the test properly functioned using the breadboard, the team then repeated the test in the same pattern with our first barebones PCB. Rather than initially seeing if the relay could blow the fuse, the team continued to solder on a current limiting resistor and a LED to prove the relays are properly flipping and the battery is supplying power to the LED. Figure 4.1.3 displays how the new breadboard was set up prior to testing.

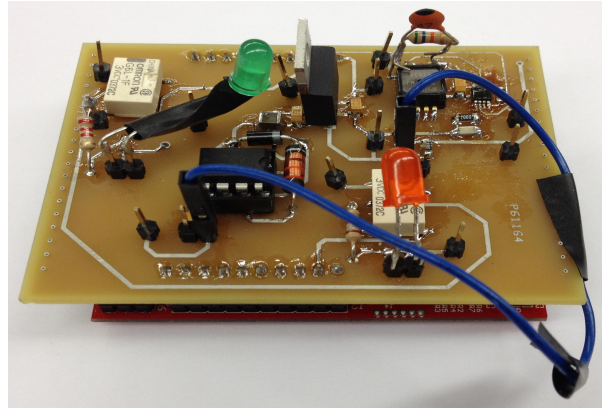


Figure 4.1.3 Barebones with LED

Unfortunately, as stated prior, this test is very uncontrolled as it is impossible to repeat this process repetitively. Although we do not know the exact change in pressure, it became our most common application test with each new revision. This is because it is easy to ensure proper functionality in all of the required systems with this test and is the most flexible test in terms of where the user can implement this test.

4.2 Elevator Test

The next test sought to try and quantify how the microcontroller interpreted height. The team wrote the code to detect the peak height and then to trigger a parachute after the height had fallen by 15 of the 1024 possible ADC values. To test this code the EPD system was brought into an elevator that had 8 total floors. The test began by starting at the lowest floor and resetting the EPD system. At this point the EDP system would start looking for the maximum height reached. The EDP system was then brought to the top floor of the elevator. The elevator then started descending toward the bottom floor again, and did eventually trigger a parachute to deploy. The team was able to hear the relay audibly switching as well view the LED light up when the system triggered. The LED lighting during this test can be seen in Figure 4.2 below. The team noted the position of the elevator when this triggering happened. In this case the system triggered when the elevator reached its 3rd floor. So the EPD went up 7 floors and descended 5 floors in this test. This test was replicated to ensure consistency. The EPD system did perform the same

Once it was known that the system changed 5 floors for 15 microcontroller height units, the floor spacing was measured. The floors were measured to be spaced at 14ft apart. With this the team was able to approximate the actual microcontroller height unit into feet. This value was determined to be 4ft 8in per microcontroller height unit.

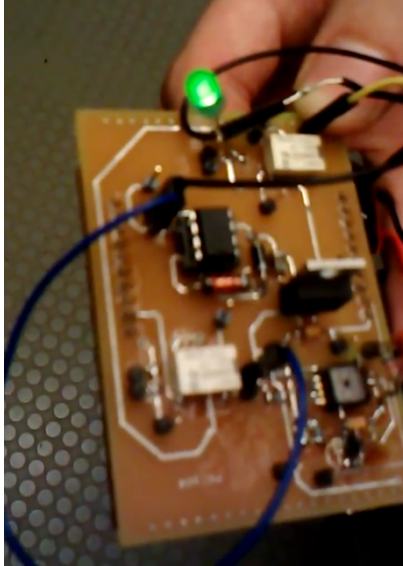


Figure 4.2 Elevator Test LED

This test showed that the basic functionality of the system was working. This test also started to quantify how our electronics were interpreting the altitude so that the EPD system could be further refined.

4.3 Vacuum Chamber Test

Previous tests conducted on the altimeter and INA showed proper functionality of the altimeter and INA. It showed the altimeter responded with a pressure change and the INA amplified and added a DC offset to its incoming signal. However, the straw test was not a controlled test meaning the variable, pressure, being changed was unknown.

A vacuum chamber was next used to test the whole system and compare its output with specific pressure. Also, the MCU was programmed to turn on the two LEDs when the parachutes would normally deploy. The vacuum chamber measures the pressure in inches of mercury. 1in of mercury is approximately equal to a change in 3.386kPa. The amount of change in pressure the team expects upon launch is about 3.487kPa, so the change in mercury is about 1in. The final system was placed inside the vacuum chamber along with a hand-held voltmeter that was used to measure the output of the INA. First the chamber was set to match the absolute pressure outside. The output of the INA read 1.546V as expected. The pressure was changed by increasing the mercury by 1in and then back down to 0in. The voltage read 2.355V at 1in, which also was expected based on the calculated results. Figure 4.3 shows the vacuum chamber setup with the measured voltage. The picture on the left shows the system and voltage at 0in. The picture on the right shows the voltage of the system just after the mercury started to drop, simulating the descent of the rocket.



Figure 4.3 Vacuum Chamber Test

It can also be seen in the picture to the right that the red LED had turned on once the mercury was reduced. This simulated the first parachute's deployment which again is expected based on the descent of the rocket. As the mercury decreased even more, eventually the second LED turned on. This allowed the team to further verify not only the system worked as expected but that the INA was outputting the expected results. Table 4.3 compares all of the INA and analog altimeter results against each other along with the expected calculated results.

	Calculated	Simulated (TINA-TI)	Measured (Pressure Chamber)
Pressure at Launch (1000ft)	97.716 kPa	-	0 in of mercury
Pressure at Max Height (2000ft)	94.213 kPa	-	1 in of mercury
Altimeter Voltage at Launch	3.939 V	3.94 V	-
Altimeter Voltage at Max Height	3.783 V	3.78 V	-
INA Voltage at Launch	1.55 V	1.55 V	1.546 V
INA Voltage at Max Height	2.312 V	2.3 V	2.321 V

Table 4.3 Analog Altimeter and INA Results

As can be seen all of the results compare closely with what is expected. Some of the error could have come from human error since a lot of the tests involved reading graphs or adjusting knobs to change pressure. Also, the absolute pressure at launch and max height were estimated, and the conversion from inches of mercury to kPa was not exact. However, with all of the possible errors, the tests proved the system works close to what was expected.

The team also felt it was necessary to conduct a saturation test. As mentioned earlier, the ADC only converts up to 2.5V. The mercury level was increased until the voltage coming out of the INA and into the MCU was past 2.5V and then lowered. The LED stayed off while the

voltage was past 2.5V. As soon as the voltage was changed to lower than 2.5V the LED turned on as expected, concluding the vacuum chamber test.

4.4 Rocket Launch Test

The EPD system was thoroughly tested in the vacuum chamber test, therefore the team moved to do a full scale test complete with rocket launch. The Madcow Torrent rocket was prepared with a G motor with 136.8Ns of impulse and the EPD system was connected to two deployment charges with .5tsp of pyrodex powder. The particular rocket motor that was purchased needed a casing, which was not purchased; therefore, substitute casing was made to house the rocket motor. When launched, the rocket motor casing did not hold perfectly, letting some of the energy escape the casing and pushed the upper half of the rocket off of the rocket prematurely. The dramatic change in the center of gravity of the lower section of the rocket, along with the pull of the cord connected to the payload section, brought the lower half of the rocket to the ground. With a failed launch, the EPD system did not detect a large enough change in altitude and therefore never cued the deployment of the parachutes. This was seen as the lone success of the launch, as the EPD system, though on and prepared to fire, did not go off at any time of the disassembly.



Figure 4.4.1 The Madcow Torrent after its launch. The purple objects are the undeployed charges.

The damage done to the rocket due to its failed launch was substantial enough to prevent further testing with it. The fins at the bottom of the rocket were slightly pushed in and the upper section of the bottom half of the rocket was no longer stiff. The most significant damage to the rocket was the motor housing. The custom housing could not be re-opened and in removing it, parts of the casing that buckled damaged the bottom of the rocket. With the prospects of relaunching the large rocket gone, the team devised a test with the medium rocket that had a successful launch previously without the EPD system.



Figure 4.4.2 The Rising Star post launch. One of the two charges deployed

The Rising Star model rocket is a medium model rocket that is not designed for an EPD system. The motors for this style of rocket generally have deployment charges built into the motor to separate the nosecone and payload from the rocket. The team did not feel comfortable tampering with the motor that was purchased with this rocket to remove the deployment charge. However, the team wanted to have a definitive test of the EPD system that would give a clear signal when the charges would detonate. The team modified the payload section of the rocket to allow for the charge wires to connect to the nosecone of the model rocket. On the nosecone, two charges were attached that had .25tsp of pyrodex, located on opposite sides of the nose cone. This launch did reach a significant change in altitude to activate the EPD system. However, the modifications to the rocket caused the rocket to begin to go sideways and then fall much sooner than expected. The built in deployment charge knocked the battery out of the EPD system, but the EPD system was able to ignite the drogue charge before it lost power. The team is confident that had the battery been secured with more than the battery holder, the main charge would have ignited.

Unfortunately, this second attempt to test the EPD system in a rocket did not yield conclusive results. Only parts of the EPD system has been fully tested in a rocket. The large rocket test proved that a significant change in altitude was needed to set off the deployment charges. The second rocket launch nearly completely confirmed the system; however, the EPD system did not maintain power long enough to deploy the main charge.

Chapter 5 - Final Cost, Schedule, Summary, and Conclusions

5.1 Cost

The total prototyping and developing cost of our final design has cost \$831.08. This cost is over our original \$500 budget. Looking at Figure 5.1, nearly a third of the prototyping cost has gone into shipping. Some of the rocket components needed for this project were physically large and some were explosive resulting in high shipping costs. Orders could have been better consolidated to reduce this cost.

The second largest cost was PCB manufacture. The team attempted to fabricate PCB prototypes through the ECE shop to reduce the cost, but a few of the components used were too small to meet the ECE shop's capabilities. The rest of the team's budget went into rocket components and electronics. These parts constituted the bulk of our actual budget and project.

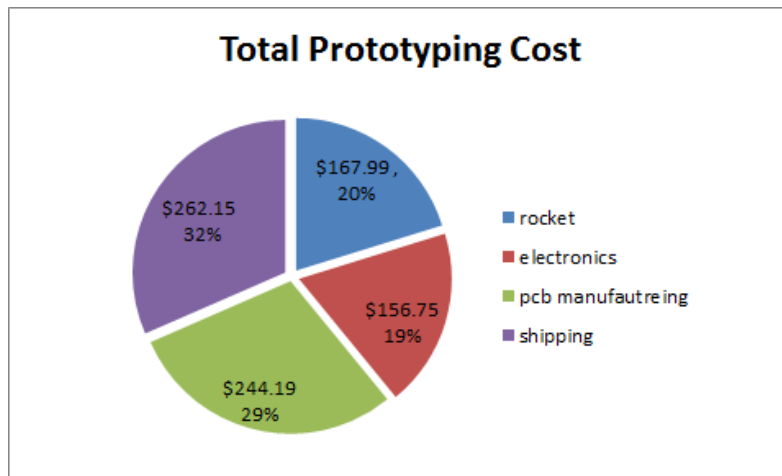


Figure 5.1 Total Prototyping Cost

While the team did go over budget, there were three prototype revisions made to our overall design. The team also conducted three rocket launches with two different rockets. Creating revisions was costly, but allowed the team to create a refined product. The last revision of the design should be delivered and built by design day, and is near a commercial quality product.

5.2 Schedule

Milestone	Expected Completion Date	Date completed
Sponsor Mini Project	Feb. 12, 2014	Feb. 17, 2014
Research Components	Feb. 17, 2014	Feb. 25, 2014
Build Rockets	Apr. 24, 2014	Mar. 5, 2014
Test Sensors	Mar. 19, 2014	Mar. 14, 2014
Test Analog Circuitry	Apr. 2, 2014	Mar. 16, 2014
Test Power System	Feb. 21, 2014	Mar. 19, 2014
Test Deployment System	Mar. 26, 2014	Apr. 16, 2014
Test Ignition system	Mar. 19, 2014	Mar. 7, 2014
Final launch attempt	Apr. 23, 2014	Apr. 21, 2014

Table 5.1 Schedule of major milestones

The team was able to complete the project in the time allotted for the project. Due to the fluidity of the design approach the team took, some scheduled milestones were reached ahead or behind schedule. This did not impede the progress of the project. Some items would have been accomplished sooner or with higher quality if more research had been done on all parts of the project. This would have led to more conclusive decisions on what parts would be ordered and when tests could begin.

5.3 Future Work

In the future, the team would like to see further testing of the EPD system and expanded capabilities. The team did not have a successful launch with a large model rocket; therefore it would be ideal to test the EPD system in a large rocket to see the product work as intended. A test with the BoosterPack system as well as the monolithic design would be desired. These two PCB designs should function in the same manner, so it would be good to see them function the same way.

Once the EPD system has been tested, there are a few modifications that the team would desire to make. The first is to find a way to report flight data afterward. This can be done with external memory such as flash or a SD card, a LCD display, or a buzzer that beeps morse code. The second modification would be to decrease the weight of the EPD system by using a lighter battery.

5.4 Conclusions of Design

The team was requested by Texas Instruments to design and demonstrate an EPD system using components made by TI. The final design of the EPD system was mostly a success. The system passed all of the bench tests that demonstrated the functionality of the system. The test launches, although not ideal in their result, did demonstrate the functionality of the EPD system as well. More time could have been devoted to research of materials for the EPD system and launching rockets. This potentially would reduce shipping cost and extraneous costs, as well as allowed testing to begin earlier. The successes that were had this semester would not have been possible with the abundance of testing that was done. Documenting all of these tests also allowed for multiple revisions of the system and the creation of more tests. The combination of all the test demonstrates a working EPD system, even in the absence of a successful rocket launch.

One of the failures of the semester was that not all design decisions were carefully researched. Researching topics both informs those working on the project of what ideas can work and can prevent bad purchases. One issue that was encountered over the course of the semester was the feeling that the project was behind schedule. Most of the major deadlines for the project were met. Some aspects, like the launching of the rockets, were weather dependent and outside of the team's locus of control. Other aspects, such as the completion and verification of code, were only dependent on the team's availability. However, at times, parts of the team were working on aspects of the project that would lead to a dead end. These dead ends could have been prevented if more attention was given to the datasheet, as other factors play a part in whether an idea will work. Therefore, a large takeaway is that doing good research saves time and money.

One of the things that helped make the project successful was the amount of testing that was done. Each subsystem was tested to confirm its functionality and confirm that it would interface properly with other subsystems. This allowed for some systems to be changed before final testing. Having a multitude of tests also allows for having data to support claims and prove functionality when other data points are missing. Even though the EPD system has not been tested successfully in the large rocket, it has been tested in many ways that shows that it would work if a successful launch occurred.

The third takeaway lesson from this project has been the importance of clear communication in teams. This communication can affect many aspects of design. Over the course of the project, breakdowns in communication led to issues in completing testing to validate the design of a particular subsystem. This led to increased tensions at times, wasted budget, and delay of the project.

There were examples of good communication. When testing, the team would be very thorough in explaining the kind of tests that would be done and verifying that it would be helpful to the project. When intentions and ideas are communicated well by the team, the team is able to use their expertise to improve the design and provide better explanations for design decisions for inquiries or reports.

Appendix 1 - Technical Roles, Responsibilities, and Work Accomplished

Michikazu Aono



Michikazu's technical role on the project was to assist in the assembly and verification of subsystem functions of the prototype EPD system, research of viable components to use in the sub systems and prepare the test rockets for launch. The project provided an opportunity to work in a team of members with diverse background in technical skills, experience in dealing with prototyping a product, software programming and hardware simulation.

During the initial component research phase, time was spent on comparing the specifications and determining the different types of analog and digital altimeter available on the market. While investigating viable altimeters, it was found many of these devices are on backorder, discontinued or too small for the team to use. Also during this phase the TI EZ430 Chronos watch was also brought in as a solution choice.

Once PCBs were printed and component orders had arrived, the member assisted in soldering on the components to the PCB and verifying each subsystem functioned as designed. Initial component/system checks were performed in the senior design lab using power supplies, function generators, and oscilloscope. During testing, oscilloscope screens were captured and shared with the rest of the team for analysis at a later time.

In order to ensure the system could be tested in a model rocket, EPD system housings were fabricated for the two type of rockets that would be used along with several other components essential for launch. The RisingStar (smaller rocket) received a small housing fabricated out of carbon fiber and epoxied together, which would house the electronics and battery. The Torrent (larger rocket) also received a custom payload tray along with major repairs conducted due to fire damage from the previous team that used the rocket. A rocket motor housing was machined, last minute due to purchasing errors, to allow for testing the system. Lastly a low budget alternative launch rail was developed to allow for launching the larger rocket.

Max Cooper



Max's technical role on the project was to design a PCB to incorporate the sensors, MCU, and power system. Prior to completing the main PCB for the EPD system, our sponsor assigned a mini op-amp project to the team. This opportunity helped refine max's PCB skills in learning how to create a professional schematic and layout. The team had the ECE shop fabricate the PCB and he was then able to test the circuit to ensure proper functionality.

After completing the mini project Max assisted the team in determining components for the project PCB. He determined the necessity of an added linear voltage regulator for a 3.3V rail to output power to the MCU. Another section he completed for the project was how to deploy the parachutes. Max found a relay that would draw enough current from the MCU to energize the coil inside the relay to flip the relay. He picked the Omron G6L-1F because the relays have a rated voltage of a 3 VDC signal and the MCU

outputs as high as a 3.6 V signal. Therefore the relay was able to operate without any faulty switching.

Once the team completed the initial parts selection Max began creating the schematic for the project PCB. He created several new symbols for the altimeter, instrumentation amplifier, and relays. After Max completed the schematic he moved on to the laying out the components. The layout required the PCB be sized compact enough to fit on top of the Launchpad and fit the size constraints required for the payload bay in the rocket. Throughout the project Max completed four revisions and the team fabricated three versions to ensure proper functionality with modified components.

Another small technical contribution to the EPD system Max helped during the project was the development of the code to deploy the parachutes. He was also actively involved in other portions of the project that needed to interact with the code. By adding input with these stages of the design, he was more aware of design issues that would affect the code for outputting signals to the MCU.

The final role Max assisted with was helping program the MCU for determining the input signal from the MCU analog-to-digital converter. This permitted the team to be able to ensure the MCU could detect the peak altitude by noticing a change from the instrumentation amplifier.

Zane Crawford



Zane's technical contribution to the EPD system was the development of the code to deploy the parachutes. He was actively involved in other portions of the project that needed to interact with the code. By having input with these stages of the design, Zane was more aware of design issues that would affect the code.

The first system that he helped develop was the amplification of the analog altimeter. It was known that the output of the analog altimeter would need signal conditioning. Zane was given the task to explain to the group the different options for the amplification of the signal. With encouragement from Peter Semig, he learned and presented the advantages of using an instrumentation amplifier over an operational amplifier for the EPD system, which was incorporated into the final design. After revising the circuit to operate in the desired range and simulating the INA326 in TI-Tina, Zane was given the task of working on the digital altimeter.

Zane worked on making the digital altimeter communicate with the MSP430G2553 using an SPI interface. He was given the task of creating code for the digital altimeter along with Huang. Zane investigated how SPI is implemented on the MSP430 and worked to make sure that the microcontroller correctly interpreted the digital altimeter output. He was nearly successful in implementing SPI on the MSP430G2553. However, not all the values were being correctly stored, despite the ordering of events appeared correct on the oscilloscope. The digital altimeter was removed from the design to keep on schedule and he was transitioned to the main code as well as testing the parachutes.

Zane also helped write the main code for the EPD system. The basic algorithm is explained in chapter three of this report. He helped make sure that the code was not susceptible

to misfiring due to noise in the ADC's input. Zane also wrote the code to store data in flash and investigated methods to retrieve this data from the microcontroller. Once the code was in a state that it could be tested, he helped test the amount of pyrodex needed to deploy the parachutes.

Qiaosen Huang



Qiaosen's main technical role on this project was to debug and program the MSP430 microcontroller to work with the altimeters and supplemental circuits on board. Initially, he was working on the digital altimeters, researching how to communicate with the digital altimeter through SPI. He was converted a code originally for a different, non-TI, MCU into what a MSP430 can understand. This task proved to be fairly difficult and consumed a considerable amount of time to debug. After not yielding correct results, he assisted in investigating solutions to the coding problem until the team decided to refocus the project.

Huang initially worked on making a main code for the MSP430F2013. Before developing a complete code that would function as a EPD system, he tested the ADC of the MCU. This would prevent making and testing the code with a non-working system. He discovered that the ADC was not behaving as expected and the the ADC was not able to register the changes. Huang then researched sigma-delta ADC and the implementation of the ADC to find what would be needed to make the system function. He, along with other team members, presented a report of the findings and made recommendations of what could be done to continue the project. From this meeting, the team worked exclusively with the MSP430G2553.

Huang was then tasked with creating test and main codes for the analog altimeter. He helped develop a test code that would read the analog altimeter's output directly. This code accounted for the noise on the output by counting the number of decreasing values. It would count decreasing values as the analog altimeter has an inversely proportional relationship between pressure and voltage. This was done before the straw test was refined, making verification slightly more difficult. For the main code, Huang researched methods for storing data from the launch and accessing that data after the launch.



Pedro Rodriguez

Pedro's technical role in the project was to develop, test, and assemble the electronics of the EPD system. He started the semester by developing the conceptual level designed and broke the problem up into separate sub-systems (sensor/microcontroller/power system/ ignition system). He assisted other team-members in component selection and system testing.

Once the project was outlined Pedro's primary focus was work on the power and ignition systems. He developed and tested 3 different ignition system. The first system used an inductor as an energy storage device and then released the store energy it the ignition fuse. The Second system was similar except a capacitor was used for a storage device. The third system was simply using the battery and a current limiting resistor to provide the energy to start the igniters.

The power system designed by Pedro underwent several revisions to ensure a good compromise between stability, cost, size, and efficiency. The final design incorporates a buck converter to take the supplied battery source and create a 5V power supply. There is also a linear voltage regulator that runs off the 5V supply that will then produce a 3.3V source to run the microcontroller. The power supply system was toughly tested under a variety of conditions to ensure the system will reliably operate within all the other components during a rocket launch. One example test was looking at how the power system robustness when the high current draw from the igniters is pulled from the battery.

Another major technical contribution that Pedro made was to revise and recreate the final PCB design in Eagle. This final revision is meant to be a stand-alone board which is different than the team's previous BoosterPack approach to the PCB. To do this the PCB had to be completely reworked in the new program. New symbols and part footprints had to made for all major components with the exception of the microcontroller. This PCB also refined the overall look of the PCB to include the Texas Instruments logo as well as include terminal connectors to use for off board connections. This PCB design was made to better make out EPD system look like other commercially available options.

Jessica Shelby



Jessica's technical role for the project was to debug any problems the team was experiencing. She first worked with the team in researching altimeters and ended up finding the digital altimeter with the breakout board that the team decided to use. The team's sponsor had each individual member complete a mini project. Through this she learned how to run simulations in TINA-TI. She also modeled and ran simulations for both the op amp and INA used throughout the project.

Most of the debugging came from bread boarding individual sections and also going through the PCB. Jessica breadboarded the INA with the altimeter as its input and power coming from the power section to verify functionality. The output was correct but with minor noise due to the sensitivity to parasitic capacitance when being bread boarded. She had a couple of the passive components soldered onto the actual pins which reduced the noise.

Jessica assisted another team member on debugging the code of the 16-bit ADC microcontroller when they realized it had a sigma-delta differential ADC which cut the voltage range in half, not providing the reference voltage we needed. The team then switched

microcontrollers. The new microcontroller was experiencing problems with turning on the relays, but another team member and she were able to prove that the output pin did indeed have enough power to turn on the relay and thus the problem narrowed in on the code.

One of the final system tests involved using a vacuum chamber to test the final functionality and compare the voltage output under controlled pressure. After the instrumentation amplifier was chosen and designed Jessica researched and performed calculations to see what the expected voltage results should be at certain altitudes for the vacuum chamber test. She assisted others in running the final tests and also checking the response of the system under saturation.

Appendix 2 - References

"Air Pressure and Altitude above Sea Level." *The Engineering Toolbox*. N.p.. Web. 22 Apr 2014. <http://www.engineeringtoolbox.com/air-altitude-pressure-d_462.html>.

"Convert Inches of Mercury to Kilopascals." *Kyle's Converter*. N.p., n.d. Web. 22 Apr 2014.
<<http://www.kylesconverter.com/pressure/inches-of-mercury-to-kilopascals>>.

Freescale Semiconductor, "Media Resistant Integrated Silicon Pressure Sensor for Measuring Absolute Pressure, On-Chip Signal Conditioned, Temperature Compensated and Calibrated," MPXHZ6116A Series datasheet [Rev. 2 June 2010].
<http://cache.freescale.com/files/sensors/doc/data_sheet/MPXHZ6116A.pdf?pspl=1&Parent_nodeId=1360710550819737433993&Parent_pageType=product>.

Measurement Specialties, "MS5607-02BA03 Barometric Pressure Sensor, with stainless steel cap", Sep 2012. <<http://www.meas-spec.com/downloads/MS5607-02BA03.pdf>>.

Omron, "Extremely ThinS PST-NO Flat Relay, One of the Thinnest Relays in the World", G6L datasheet, Sep 2011.
<[http://components.omron.com/components/web/pdflib.nsf/0/51A3390695AC774885257201007DD58D/\\$file/G6L_0911.pdf](http://components.omron.com/components/web/pdflib.nsf/0/51A3390695AC774885257201007DD58D/$file/G6L_0911.pdf)>.

Parallax Inc, "Altimeter Module MS5607 (#29124)", Parallax Altimeter Module MS5607 Datasheet, Nov 2011. <<http://www.parallax.com/downloads/altimeter-documentation>>.

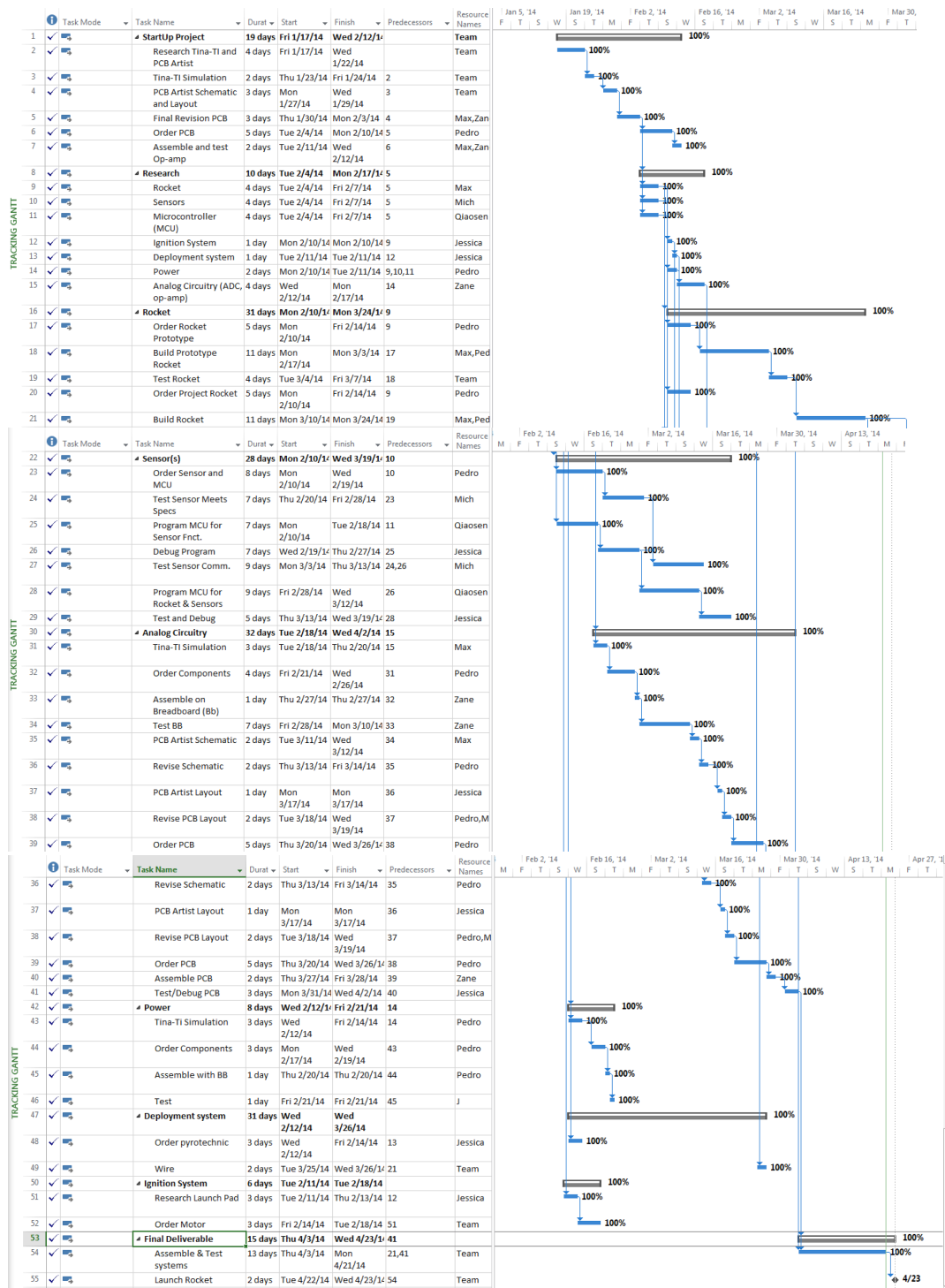
Parallax Inc, "Altimeter Module MS5607 - Top View". Parallax MS5607
<http://www.parallax.com/sites/default/files/styles/full-size-product/public/29124_0.png?itok=Ra5B4HAq>.

Texas Instruments, "0.05uV/C max, SINGLE-SUPPLY CMOS OPERATIONAL AMPLIFIERS Zero-Drift Series," OPA335 datasheet, June 2002 [Revised July 2003].
<<http://www.ti.com/lit/ds/symlink/opa335.pdf>>.

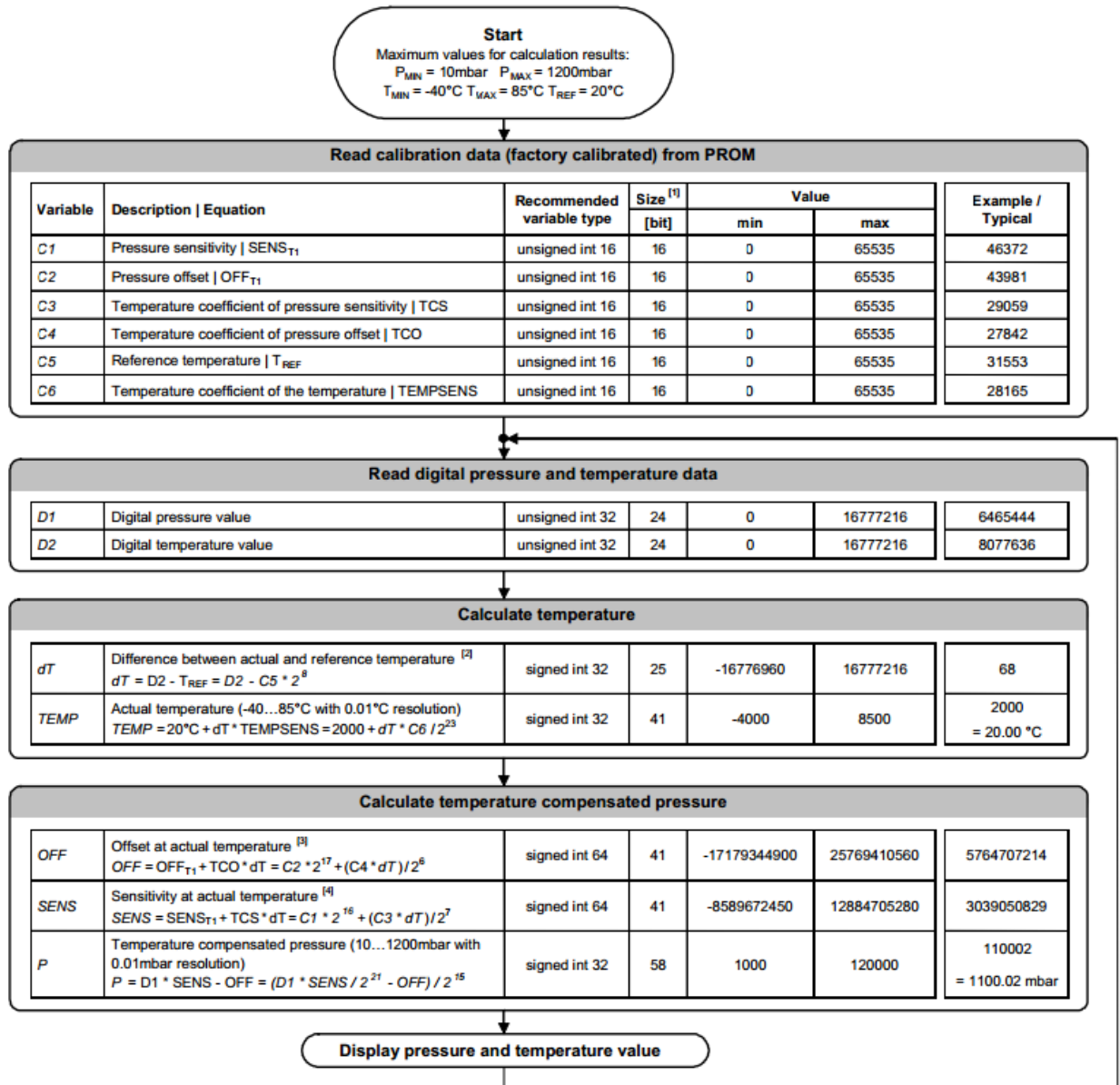
Texas Instruments, "Precision, Rail-to-Rail I/O INSTRUMENTATION AMPLIFIER," INA326 datasheet, Nov. 2001 [Revised Nov. 2004]. <<http://www.ti.com/lit/ds/symlink/ina326.pdf>>.

Appendix 3 - Code, Schematics, and Additional Pictures

Appendix 3.1 Gantt Chart



Appendix 3.2 Pressure Computation Process



Notes

- [1] Maximal size of intermediate result during evaluation of variable
 [2] min and max have to be defined
 [3] min and max have to be defined
 [4] min and max have to be defined

```

/*
 * main.c
 *
 * Created on: March 10, 2014
 * Author: MSU Team 11
 */

#include <msp430g2553.h>

void main(void){

    WDTCTL = WDTPW + WDTHOLD; // stop watchdog timer

    BCSCTL2=0x36; //ACLK = VLO
    BCSCTL3=0x20; // MCLK = DCO/8 SMCLK = DCO/8

    P1REN = 0x08;
    P1DIR = BIT6+BIT1; //output drogue chute on P1.1 (use 1.6 led as well)
    P1SEL = 0x00;
    P1OUT &= 0x00; //make sure output to relays is clear

    P2DIR = BIT6; //output main chute on P2.6
    P2SEL = 0x00;
    P2SEL2 = 0x00;
    P2OUT &= 0x00; //make sure output to relays is clear

    ADC10CTL1 = 0x0004; //use pin P1.0 for input
    ADC10AE0 = 0x01; // enable pin a0
    ADC10CTL0 = SREF_1+REF2_5V + REFON + ADC10ON + MSC;// use reference of 2.5 V and
turn on sampling with 4cycle hold
    _delay_cycles(5); // pause

    ADC10CTL0 |= ENC + ADC10SC; //enable adc10

    int delay; //delay variable
    volatile long voltRange,basevolt,currvolt,prevvolt1,vmax;
    volatile long drogueDeploy, mainDeploy; //variables for storing data
    volatile long up,down,drogue,mainch; // variables for loops

    up=0; //variable to check if launched
    down =0; //variable to check if falling
    drogue =0; //variable to check when to deploy drogue chute
    mainch=0; // variable to check when to deploy main chute

    // voltage= (ADC10MEM)*2.5/1023.; conversion of ADC val to voltage

    basevolt=ADC10MEM; //base voltage
    currvolt=basevolt; //initialize value
    prevvolt1=basevolt; //last voltage seen
    vmax=basevolt; //maximum voltage (at peak)
    drogueDeploy=basevolt; //initialize value
    mainDeploy=basevolt; //initialize value
    voltRange=0; //initialize value

    while(!up){ //loop to poll for launch

```

```

    currvolt = ADC10MEM; //grab value from ADC
    if(currvolt > (basevolt+25)){up=1;} //if greater than the measured steady-
state error, say launched
}
while(up){ //loop to find peak
    currvolt = ADC10MEM; //grab adc value
    if(currvolt<(vmax-25)){ //if definitely lower than the max
        up=0; //now descending
        down=1;
    }
    else{ //keep waiting for apogee
        prevvolt1=currvolt;
        if(prevvolt1>vmax){vmax=prevvolt1;} // always take the highest voltage
possible
    }
}
voltRange =vmax-basevolt; //calculate range of flight

while(!drogue){// look for the specified height for the drogue chute
    currvolt = ADC10MEM;
    //if(currvolt < (vmax-30)){ //deploy drogue near apogee
    if(currvolt < (voltRange*.7+basevolt)){ //deploy drogue at 70% of altitude
        //output voltage
        P1OUT|=BIT1 +BIT6; //trigger fuse 1 (and green LED on BoosterPack
        drogue = 1; //end the loop
        drogueDeploy = currvolt;
    }
}

while(!mainch){ // look for the specified height for the main chute
    currvolt = ADC10MEM;
    //if((currvolt <(voltRange/2+basevolt) )){ //deploy at 50% of altitude
        if((currvolt <(voltRange*.4+basevolt) )){ //deploy at 40% of altitude
            for(delay=0; delay<10000; delay++); // pause before turning off fuse 1
            P1OUT&=~BIT1 +~BIT6; //turn off fuse 1
            P2OUT|=BIT6; //turn on fuse 2
            mainDeploy=currvolt;
            mainch = 1; // end the loop
        }
        for(delay=0; delay<10000; delay++); //pause before turning off fuse 2
        P2OUT&=~BIT6; //turn off fuse 2
    }
}
}

```

```

#include <msp430.h>

/*
 * main.c
 */

#include <msp430g2553.h>
#include <stdint.h>
/*
 * main.c
 */
/* MOSI   P1_7
 * MISO   P1_6
 * SCK    P1_5
 *
 *
 * SPI rate = 2mhz.
 */

#define F 2000000
#define CYCLE_PER_MS F/1000
#define CYCLE_PER_US F/1000000

void spi_init(void);
void spi_write(unsigned int);
unsigned int spi_read(void);
void cslow(void);
void cshigh(void);
void digital_reset(void);
unsigned int digital_readcoef( unsigned int);
unsigned long digital_read(unsigned int);
unsigned crc4(unsigned int n[]);
volatile unsigned char RXD, TXD;
volatile unsigned long D1, D2;
unsigned int c[8];
volatile double P, T, dT, offset, SENS;
volatile unsigned n_crc;
volatile unsigned int tp, counter=0;

volatile double oldP=0, oldT=0, oldD1, oldD2;
void _delay_ms(unsigned int s) {
    while(--s) {_delay_cycles(CYCLE_PER_MS);}
}
void _delay_us(unsigned int s) {
    while(s--) {_delay_cycles(CYCLE_PER_US);}
}

int main(void) {
    WDTCTL = WDTPW | WDTHOLD;    // Stop watchdog timer
    DCOCTL = CALDCO_16MHZ;       // master clock 16mhz
    BCSCTL1 = CALBC1_16MHZ;      // ACLK

    WDTCTL = WDTPW + WDTHOLD;    // Stop watchdog timer

```

```

P1OUT = 0x00; // P1 setup for LED & reset output
P1DIR |= BIT0 + BIT5 + BIT6; //
P1SEL = BIT1 + BIT2 + BIT4;
P1SEL2 = BIT1 + BIT2 + BIT4;

/*
P1DIR = BIT4+BIT7+BIT5+BIT6+BIT0;
P1OUT = BIT4 + BIT0;
P1SEL = BIT5+BIT6+BIT7;
P1SEL2 = BIT5+BIT6+BIT7;
*/
spi_init();

unsigned int j;
digital_reset();
for (j=0;j<8;j++) {
    c[j]=digital_readcoef(j);
}
//n_crc=crc4(c);
while(1) {
    D1=digital_read(0x00);
    D2=digital_read(0x10);
    dT=D2-c[5]*256;
    offset=c[2]*131072+dT*c[4]/64;
    SENS=c[1]*65536 + dT*c[3]/128;
    P=(D1*SENS/2097152-offset)/32768/100;
    T=(2000+dT*c[6]/8388608)/100;
    if (P<oldP) {
        // P1OUT |= BIT6;
    }
    oldP=P;
    if (D1!=oldD1 || D2!=oldD2) {
        // P1OUT |= BIT0;
    }
    oldD1=D1;
    oldD2=D2;
    //_delay_ms(200);
    //P1OUT &= ~(BIT6+BIT0);
    tp=1+1;
}

return 0;
}

void spi_init(void) {

/* UCSWRST */
UCB0CTL1 = UCSWRST;

UCA0CTL0 |= UCCKPL + UCMSB + UCMST + UCSYNC; // 3-pin, 8-bit SPI master
UCA0CTL1 |= UCSSEL_2; // SMCLK
UCA0BR0 |= 0x02; // /2
UCA0BR1 = 0; //
UCA0MCTL = 0; // No modulation

```

```

        UCA0CTL1 &= ~UCSWRST;
/* USCI B0 */
//UCB0CTL0 &=0x00;
//UCB0CTL0 =0x29;
//UCB0CTL0 |= ~UCCKPL + ~UCMSB + UCMST + UCSYNC; // 3-pin, 8-bit SPI master
//UCB0CTL1 |= UCSSEL_2; // SMCLK
//UCB0BR0 |= 8; // /2
//UCB0BR1 = 8;
//UCB0MCTL = 0;

// UCB0CTL1 &= ~UCSWRST;
return;
}
void spi_write(unsigned int s) {
//    IE2 |=UCB0TXIE;
    while (!(IFG2 & UCA0TXIFG));
//while (!(UCB0TXIE & UCB0TXIFG));
    UCA0TXBUF = s;
    while (!(IFG2 & UCA0TXIFG));
//    IE2 &=~UCB0TXIE;
//    IFG2 &=~UCB0RXIFG;
    return;
}
unsigned int spi_read(void) {
//UCB0TXBUF =0x00;
    unsigned int tmp;
//    IE2 |=UCB0RXIE;
    while (!(IFG2 & UCA0TXIFG));
//while ( !(UCB0TXIE & UCB0TXIFG)) );
    UCA0TXBUF = 0x00;
    while (!(IFG2 & UCA0RXIFG));
//while ( !(UCB0RXIE & UCB0RXIFG)) );
    tmp = UCA0RXBUF;
//    IE2 &=~UCB0RXIE;
    return tmp;
}
void cslow(void) {
    P1OUT &= ~BIT5;
    P1OUT |=BIT0;
    return;
}
void cshigh(void) {
    P1OUT |= BIT5;
    P1OUT &=~BIT0;
    return;
}
void digital_reset(void) {
    cshigh();
    _delay_ms(1);
    cslow();
    spi_write(0x1e); //reset cmd
    _delay_ms(50);
}

```

```

        cshigh();
        _delay_ms(5);
        return;
    }

    unsigned int digital_readcoef( unsigned int c) {

        unsigned int tempd,temp2;
        unsigned int temp;
        temp2=0xFF;
        cslow();
        spi_write(0xA0+c*2);
        temp=spi_read();
        if(temp >=0xF0) P1OUT ^= BIT6;
        tempd=temp*256;
        temp=spi_read();
        tempd+=temp;

        _delay_ms(50);
        cshigh();
        _delay_ms(5);
        return tempd;
    }

    unsigned long digital_read(unsigned int c) {

        unsigned volatile ret;
        unsigned long volatile temp=0;
        cshigh();
        cslow();
        spi_write(0x40+c);
        switch (c & 0x0f) {
            case 0x00: _delay_ms(50);break;
            case 0x02: _delay_ms(3);break;
            case 0x04: _delay_ms(4);break;
            case 0x06: _delay_ms(6);break;
            case 0x08: _delay_ms(10);break;
        }
        P1OUT ^=BIT0;
        ret=spi_read();
        ret=spi_read();
        temp+=ret*65536;
        ret=spi_read();
        temp+=ret*256;
        ret=spi_read();
        temp+=ret;

        cshigh();
        _delay_us(900);
        return temp;
    }

```

```

unsigned crc4(unsigned int n[]) {
    int cnt;
    volatile unsigned int n_rem;
    unsigned int crc_read;
    unsigned int n_bit;
    n_rem = 0x00;
    crc_read=n[7];
    n[7]=0xff00 & n[7];
    for (cnt=0;cnt<16;cnt++) {
        n_rem ^= (cnt%2==1?(unsigned short)(n[cnt>>1] & 0x00ff):(unsigned
short)(n[cnt>>1]>>8));
        for (n_bit=8;n_bit>0;n_bit--) {
            if (n_rem & 0x8000)
                n_rem=(n_rem<<1)^0x3000;
            else
                n_rem=n_rem<<1;
        }
    }
    n_rem=0x000f & (n_rem>>12);
    n[7]=crc_read;
    return n_rem ^ 0x00;
}

```